APPLIED CYBER OPERATIONS
CAPSTONE PROJECT REPORT

REACTIVE AGGREGATE MODEL PROTECTING AGAINST REAL-TIME THREATS

by

Kenneth G. Baugess
Jason R. Chamberlain
Samuel K. Chung
Ryan F. Kelly

September 2014

Thesis Advisor:	Shelley Gallup
Co-Advisor:	Thomas S. Anderson
Second Reader:	Scott McKenzie

THIS PAGE INTENTIONALLY LEFT BLANK

REPORT DOCUMENTATION PAGE			*Form Approved OMB No. 0704–0188*

1. AGENCY USE ONLY *(Leave blank)*	**2. REPORT DATE** September 2014	**3. REPORT TYPE AND DATES COVERED** Master's Thesis

4. TITLE AND SUBTITLE REACTIVE AGGREGATE MODEL PROTECTING AGAINST REAL-TIME THREATS	**5. FUNDING NUMBERS**
6. AUTHOR(S) Kenneth G. Baugess, Jason R. Chamberlain, Samuel K. Chung, Ryan F. Kelly	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) Naval Postgraduate School Monterey, CA 93943-5000	**8. PERFORMING ORGANIZATION REPORT NUMBER**
9. SPONSORING /MONITORING AGENCY NAME(S) AND ADDRESS(ES) N/A	**10. SPONSORING/MONITORING AGENCY REPORT NUMBER**

11. SUPPLEMENTARY NOTES The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the U.S. Government. IRB Protocol number ____N/A____.

12a. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited	**12b. DISTRIBUTION CODE**

13. ABSTRACT (maximum 200 words)

Current network protection technologies often require code recompilation to integrate new technologies, can be prone to denial of service attacks, may require invasive software applications to provide an automated response, and provide little to no protection against unknown threats. Unknown threat discovery generally requires an expert human analyst in an impractical labor-intensive process, but these analysts are in short supply. A Vector Relational Data Modeling approach was implemented to automate the human-intensive decision-making and subsequent response processes when a common phpMyAdmin attack is suspected. We modeled constituent component technologies and data sources within the Global Information Network Architecture, a DOD network certified information modeling framework, and constructed a cyber test range consisting of multiple servers. This implementation and testing of Reactive Aggregate Model Protecting Against Real-time Threats demonstrated the successful employment of an information apparatus that executed the complex processes necessary to mitigate phpMyAdmin cyber threat detection and response.

14. SUBJECT TERMS information modeling, decision-making, Vector Relational Data Modeling (VRDM), cyber threat detection, phpMyAdmin, Global Information Network Architecture (GINA), knowledge management, multi-criteria decision analysis	**15. NUMBER OF PAGES** 139
	16. PRICE CODE

17. SECURITY CLASSIFICATION OF REPORT Unclassified	**18. SECURITY CLASSIFICATION OF THIS PAGE** Unclassified	**19. SECURITY CLASSIFICATION OF ABSTRACT** Unclassified	**20. LIMITATION OF ABSTRACT** UU

NSN 7540–01-280-5500 Standard Form 298 (Rev. 2–89) Prescribed by ANSI Std. 239–18

THIS PAGE INTENTIONALLY LEFT BLANK

REACTIVE AGGREGATE MODEL PROTECTING AGAINST REAL-TIME THREATS

Kenneth G. Baugess   Jason R. Chamberlain
Samuel K. Chung   Ryan F. Kelly

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN CYBER SYSTEMS AND OPERATIONS

from the

NAVAL POSTGRADUATE SCHOOL
September 2014

Authors:	Kenneth G. Baugess Jason R. Chamberlain Samuel K. Chung Ryan F. Kelly
Approved by:	Dr. Shelley Gallup Thesis Advisor
	Dr. Thomas S. Anderson Co-Advisor
	Scott McKenzie Second Reader
	Dr. Cynthia Irvine Chair, Cyber Academic Group

THIS PAGE INTENTIONALLY LEFT BLANK

ABSTRACT

Current network protection technologies often require code recompilation to integrate new technologies, can be prone to denial of service attacks, may require invasive software applications to provide an automated response, and provide little to no protection against unknown threats. Unknown threat discovery generally requires an expert human analyst in an impractical labor-intensive process, but these analysts are in short supply. A Vector Relational Data Modeling approach was implemented to automate the human-intensive decision-making and subsequent response processes when a common phpMyAdmin attack is suspected. We modeled constituent component technologies and data sources within the Global Information Network Architecture, a DOD network certified information modeling framework, and constructed a cyber test range consisting of multiple servers. This implementation and testing of Reactive Aggregate Model Protecting Against Real-time Threats demonstrated the successful employment of an information apparatus that executed the complex processes necessary to mitigate phpMyAdmin cyber threat detection and response.

THIS PAGE INTENTIONALLY LEFT BLANK

TABLE OF CONTENTS

LIST OF FIGURES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF TABLES

THIS PAGE INTENTIONALLY LEFT BLANK

LIST OF ACRONYMS AND ABBREVIATIONS

ACL	Access Control List
CentOS	Community Enterprise Operating System
CLI	Command Line Interface
CRREL	Center Cold Regions Research and Engineering Laboratory
CSO	Cyber Systems and Operations
DISE	Distributed Information and Systems Experimentation
DIACAP	Department of Defense Information Assurance Certification and Accreditation Process
DMZ	Demilitarized Zone
DNS	Domain Name System
DOD	Department of Defense
DoS	Denial of Service
DPIMS	Dragon Pulse Information Management System
EAI	Enterprise Application Integration
EII	Enterprise Information Integration
ERDC	Engineering Research and Development
ETL	Extraction, Transformation, and Loading
FTP	File Transfer Protocol
Gb	Gigabit
GB	Gigabyte
GHz	Gigahertz
GINA	Global Information Network Architecture
GUI	Graphic User Interface
IDS	Intrusion Detection System
IIS	Internet Information Services
IP	Internet Protocol
IPS	Intrusion Prevention System
IPSec	Internet Protocol Security

ITACS	Information Technology and Communications Services
KM	Knowledge Management
LTS	Long Term Support
MCDA	Multi-Criteria Decision Analysis
MCDM	Multi-Criteria Decision Making
MS	Microsoft
NIC	Network Interface Controller
NIST	National Institute of Standards and Technology
NOC	Network Operations Center
NPS	Naval Postgraduate School
Nmap	Network Mapper
NRC	National Research Council
ODBC	Open Database Connectivity
OLAP	Online Analytical Processing
OS	Operating System
PHP	PHP Hypertext Preprocessor
RAID	Redundant Array of Independent Disks
RAM	Random Access Memory
RAMPART	Reactive Aggregate Model Protecting Against Real-time Threats
RDC	Remote Desktop Connection
RHEL	Red Hat Enterprise Linux
RPM	Revolutions Per Minute
SAS	Serial Attached SCSI
SCSI	Small Computer System Interface
SEM	Security Event Management
SIEM	Security Incident and Event Management
SIM	Security Information Management
SOA	Service-Oriented Architecture
SOP	Standard Operating Procedure
SoS	System of Systems

SQL	Structured Query Language
SSH	Secure Shell
TCP	Transmission Control Protocol
Tor	The Onion Router
TTL	Time to Live
TTP	Tactics, Techniques, and Procedures
URL	Uniform Resource Locator
U.S.	United States
USACE	United States Army Corps of Engineers
USG	United States Government
VLAN	Virtual Local Area Network
VM	Virtual Machine
VRDM	Vector Relational Data Modeling

THIS PAGE INTENTIONALLY LEFT BLANK

ACKNOWLEDGMENTS

Team RAMPART would like to sincerely thank our Thesis Advisor Dr. Shelley Gallup. His guidance and instructions were significant to our success. We also extend our gratitude to Mr. Scott McKenzie and Dr. Thomas Anderson for their invaluable direction, counsel, and support to see us through to completion. Mr. Garett McGrath's aid was instrumental to overcome a number of technical obstacles during the course of the project. Mr. Scott Cote's and Mr. Albert "Buddy" Barreto's support enabled us to build the server assembly required for our project. Also, Mr. Ricardo Estrada provided detailed information and insight into the Naval Postgraduate School Information Technology and Communications Services procedural and operational processes. Finally, Big Kahuna Technologies, specifically Frank Busalacchi, Mike Morris, and Joe Bressler, provided extensive training and support, vital for our team to conceptually design and build a working model. By leveraging their expertise, the RAMPART team devised creative and innovative solutions to complete the capstone.

THIS PAGE INTENTIONALLY LEFT BLANK

I. INTRODUCTION

The cyber threat environment is complex and evolving necessitating that organizations take defensive measures encompassing various domains of expertise on systems and processes that must rapidly adapt to mitigate network vulnerabilities. A holistic solution to this complex, dynamic, and heterogeneous problem does not lend itself well to specialized, stove-piped, and conventional methods.

A. TASKING STATEMENT

The tasking for this capstone project was directed by U.S. Navy Fleet Cyber Command/10th Fleet, N5, Captain Roy Petty, to explore solutions in the cyber threat environment; through Dr. Shelley Gallup, director, NPS Distributed Information Systems and Experimentation (DISE) research group; Dr. Thomas S. Anderson, United States Army Corps of Engineers (USACE), Engineering Research and Development (ERDC), Center Cold Regions Research and Engineering Laboratory (CRREL); and Mr. Scott McKenzie, Research Associate, DISE research group. The project was funded by N1 through the NPS OPNAV Naval Studies Program. The tasking was to implement an information modeling solution in the Global Information Network Architecture (GINA) framework for a cyber-threat detection model; see Appendix A.

The key points in the tasking are the following:

- Develop prototype cyber attack detection capability.
- Create a unified process and behavior model for key capabilities.
- Transition from responding to detected threats to developing techniques and capabilities to hunt for and respond to potentially threatening network probes before systems have been compromised.
- Incorporate Tactics, Techniques, and Procedures (TTP) into the information apparatus to enable semi-automation of tasks and allow human operators to broaden their focus.

B. PROBLEM STATEMENT

Intrusion detection and prevention systems (IDPS) provide a limited degree of protection to computer networks. Comprehensive protection of a network requires personnel to continuously monitor network traffic for possible or real threats. Specifically, it necessitates analyzing system logs in real-time to immediately detect and respond to an attack. This human labor-intensive process is not practical for many organizations due to the high volume of network traffic. Network traffic coupled with the persistent and sophisticated nature of cyber threats increases an organization’s vulnerability to attacks. Additionally, false positives and false negatives in detecting attacks complicate an analyst’s ability to accurately identify a real attack and respond to it in a timely manner.

Adequately posturing against attacks using conventional materials and methods is resource intensive and expensive, especially for a large organization. Creating a dynamically configurable, extensible, and scalable cyber defense model will facilitate sustaining an organization’s cybersecurity posture at a reduced cost and a decreased invesent of manpower, training, and equipment. One approach to solving the resource-intensive overhead problem is by automating the human decision-making process. An efficient and effective cybersecurity system can be implemented using Vector Relational Data Modeling (VRDM) to simulate the decision process of an expert network security analyst. The RAMPART project sought to implement a cost effective, scalable, cybersecurity model capable of near real-time, semi-cognitive decision-making and automated response.

This capstone was divided into four phases. First, the team designed and implemented a proof-of-concept information apparatus based on a Naval Postgraduate School (NPS) thesis *Automated Cyber Threat Analysis and Specified Process Using Vector Relational Data Modeling* [1] that postulates that a VRDM model is capable of near real-time response to phpMyAdmin cyber attacks. Second, a cyber test range was constructed. Third, constituent technology software components were assembled for use in the extended system solution. Fourth, we tested the information apparatus against a honeypot Apache webserver using both simulated and real-world phpMyAdmin attacks. The honeypot web server served as a decoy server to collect information on the attacker [2].

THIS PAGE INTENTIONALLY LEFT BLANK

II. BACKGROUND

A. CYBER THREAT DOMAIN

The initial development of the Internet was highly focused on enabling the flow of information. Security concerns have traditionally trailed information sharing requirements, resulting in an environment that is now exploitable by cyber attackers and malware [3]. As society's dependency on the Internet grows, so do the hazards associated with its use. Cyber threats continue to grow and evolve in complexity and scale. These attacks can originate from numerous sources including other nation-states, non-state actors, criminal organizations, terrorist groups, and even unaffiliated individuals. They present a credible and dangerous risk to U.S. information and communication networks [4]. Daily attacks number in the thousands [4] and constantly bombard and stress defensive measures [5].

Cybersecurity greatly benefits from routine and frequent log analysis by identifying malicious activity, violations, and problems [6]. The National Institute of Standards and Technology (NIST) describes the problem of balancing an organization's ability to analyze logs with the continuous generation of log data as a "fundamental problem" in cyber defense [6]. Human-labor-intensive log analysis cannot keep pace with the number of attacks committed against United States Government (USG) and DOD networks. Numerous cyber events are logged daily by cybersecurity software, intrusion detection and prevention systems, network and end devices, and firewalls [6]. These logs must be filtered, analyzed, and acted upon as near to real-time as possible to limit impact of any cyber attack [7].

The Reactive Aggregate Model Protecting Against Real-Time Threats (RAMPART) system was designed to determine if an intrusion met a network administrator-defined attack criteria and if it automates any specific responses per the tactics, techniques, and procedures (TTP) of a defending organization. RAMPART was configured to execute an automated response or to alert network operators of an intrusion.

RAMPART supports cybersecurity by merging decision-making frameworks and cyber defense strategies; see Appendix C.

B. STATE OF CYBER DEFENSE

Firewalls, anti-virus software, Intrusion Detection System (IDS), and Intrusion Protection System (IPS) all contribute to the current state of cyber defenses. Their combined effects give a layered defense to computer networks. From each of these cyber defense system components, operators collect logs of the recorded events triggered by their systems as suspicious and possibly malicious activity. However, cyber attacks are rarely discovered using a single log [6], but rather through the aggregation of one or more logs. Discovering an attack is difficult because a majority of cyber defense components create thousands of pages of logs daily [6].

Conducting log analysis is problematic, partly because the type of log storage and varying log formats. Optimal configuration of the network can force cyber defense components to deliver their logs to a shared or distributed location [7]. However, configuring the network in this way adds more complexity to an already complex network. Information is stored in different orders, formats, and types. Because the formats and locations of the logs vary, the operator’s analysis is more difficult. Substantial time and effort is required of an operator to monitor, understand, and process logs while looking for often hard-to-spot details indicating an attack [6].

Commercial cybersecurity solutions exist on the market but are often more complex, expensive, and resource-intensive [6]. Often referred to as Security Information Management (SIM), Security Event Management (SEM), or Security Information and Event Management (SIEM) systems, these commercial cybersecurity solutions offer organizations the ability to aggregate data, correlate common attributes, and link user-friendly tools to create a graphic user interface (GUI) or dashboard for easy access and management [8]. Purpose-built ‘fraud detection mechanisms’ are responsible for 5 percent of all detections of intrusions against larger organizations, just behind the 8 percent for routine manual log review [9]. On average, intrusion detection systems are worse at identifying cyber threats than ad-hoc log analysis [9]. Current solutions do not adequately address the problem. A dynamically configurable, extensible, and scalable cybersecurity model is needed to protect an organization's network.

C. VECTOR RELATIONAL DATA MODELING

The team selected Vector Relational Data Modeling to assemble RAMPART because it provided a dynamic and adaptable approach to creating an information apparatus that describes and executes system behavior. VRDM allows for compatibility and interoperability of data objects between different and multiple systems. Moreover, this model supports an extensible, scalable, configurable, and reconfigurable architecture to grow a given System of Systems (SoS) ecosystem [10], [11], [12]. Specifically, VRDM enables the aggregation of disparate data and information as component objects from multiple sources, and the ability to invoke, arrange, and relate the data in a configured extensible system.

VRDM assembles component objects within the GINA framework environment through a configured specification [13], [14], [15]. VRDM permits the interoperation of data and subsequent decisions and processes to be described "as a semantically-rigorous, executable description of domain-relevant, inter-connected information objects, and then maps the semantics of the resulting information objects to the unique semantics and syntax of the underlying systems. The resulting [model was] well described, extensible, and robust" [13]. Interoperability of systems, to include individual data elements contained in any given system, creates a capability more robust than basic interactions between systems as each individual data element in a component system within the SoS is interoperable and functional in any other component system [13]. Furthermore, the extensible and universally configurable model provides an environment to define and integrate data semantics, processing logic, and information resources within and across all system domains [13]. It is clear that the VRDM approach is extremely useful, to the extent that its basis is unverified.

VRDM has three distinct layers: visual, semantic, and source. The visual layer is a representation of source records that are displayed in a web form. The semantic layer is an intermediate layer where a normalized semantic model is assembled. The source layer of VRDM connects to the data, in database tables or data streams; see Figure 1.

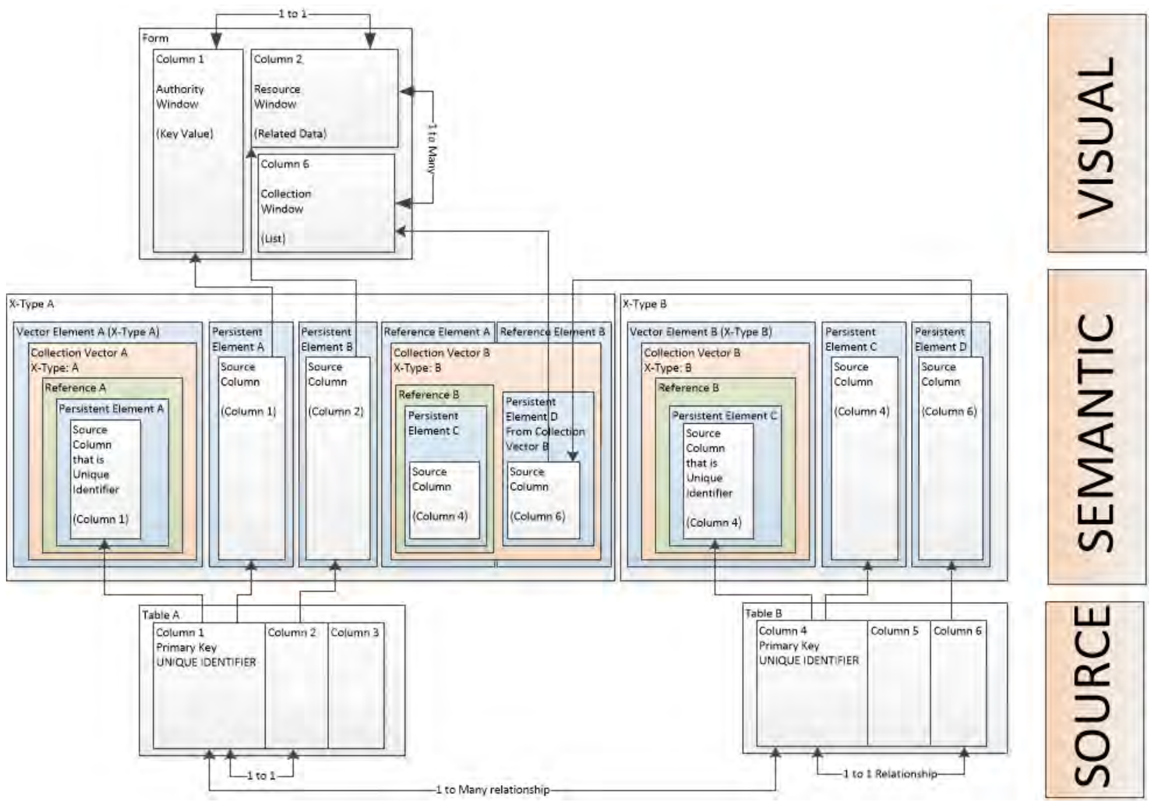

VISUAL
SEMANTIC
SOURCE
Form
Column 1 Authority Window (Key Value)
Column 2 Resource Window (Related Data)
Column 6 Collection Window (List)
1 to 1
1 to Many
X-Type A
Vector Element A (X-Type A)
Collection Vector A X-Type: A
Reference A
Persistent Element A
Source Column that is Unique Identifier (Column 1)
Persistent Element A Source Column (Column 1)
Persistent Element B Source Column (Column 2)
Reference Element A
Reference Element B
Collection Vector B X-Type: B
Reference B
Persistent Element C
Source Column (Column 4)
Persistent Element D From Collection Vector B
Source Column (Column 6)
X-Type B
Vector Element B (X-Type B)
Collection Vector B X-Type: B
Reference B
Persistent Element C
Source Column that is Unique Identifier (Column 4)
Persistent Element C Source Column (Column 4)
Persistent Element D Source Column (Column 6)
Table A
Column 1 Primary Key UNIQUE IDENTIFIER
Column 2
Column 3
Table B
Column 4 Primary Key UNIQUE IDENTIFIER
Column 5
Column 6
1 to 1
1 to Many relationship
1 to 1 Relationship

Figure 1. X-Types have Elements, relationships between X-Types, an abstraction of a database table, are specified by Vectors. Collection Vector A, and Collection Vector B shown above, relate Elements (blue boxes). The arrows indicate a configured relationship. The source, semantic, and visual layers of the model are loosely coupled, from [1].

The semantic model translates and maps all data elements within the basic structure of the SoS. This basic structure or conceptual dimension characterizes the overall SoS behavior by defining concepts, relationships, action, and components of the model; within GINA these are known as X-Types, Vectors, Services, and Elements, respectively [16]. This semantic model is decoupled from the constituent data source components allowing independent changes to be made to the model or the data sources. The data source components are integrated by normalizing databases to be compatible with individual data formats. A syntactic translator, a component of the .NET architecture, transforms different data formats. This transformation enables all data elements to communicate within the GINA environment.

Major communication protocols supported in GINA are TCP/IP, UDP, Web Services, ODBC/OLEDB, and serial communication; there is virtually no limit to number of protocols that could be supported within the VRDM information apparatus [16]. Furthermore, VRDM has implemented programmable and configurable objects to enable the complete configuration of an executable system.

There are nine core objects for the three framework layers in VRDM [1], [17]:

1. Connection: The Connection specifies the server IP address and source database name.

2. Source: The Source object specifies the Connection and data source table. The source is like a window into the remote database table where columns can be accessed as Element objects.

3. Column: The Column specifies the server database column specified by the Source object.

4. X-Type: The X-Type is an abstraction of a database table. X-Types have Elements and Vectors associated with them.

5. Element: The Element object is an abstraction of a column within a database table.

6. Vector: The Vector object forms the relationships between Elements.

7. Vector Reference: A Vector Reference is an Element assigned to a Vector that serves as a filter for values retrieved from an Element.

8. Forms: The UI page that contain the information Windows.

9. Window: The three types of Windows: Authority Window, Resource Window, and Collection Window. The Authority Window contains navigable Elements of context. The Resource Window displays the Elements in the same *row* as the Element

selected in the Authority Window. The collection Window displays related records specified by a Collection Vector.

All VRDM model objects are configured in the framework's web browser based user interface (UI). This UI is a simple, forms based web expression that utilizes text fields, check boxes, and drop down tabs for assembling the information apparatus.

VRDM leveraged with GINA's multi-dimensional architecture provided the team a way to solve the complex and challenging problem of integration, interoperability, and information exchange. Moreover, to explore and employ powerful concepts and tools to build a working model demonstrating its viability and applicability to real world problem sets.

D. SELECTED USE CASES

Team RAMPART approached the project by first selecting representative use cases of the cyber threat domain and a Network Operations Center (NOC). The selected use cases are described below:

1. PhpMyAdmin

The phpMyAdmin cyber attack is a representative use case of the cyber threat domain. The project team chose the "phpMyAdmin" attack, CVE-2010-3055, as a use case because it is an effective attack against web servers at NPS and because it demonstrates the effectiveness of a VRDM executable information apparatus in the GINA framework as a cyber threat response solution. We chose this attack because it was continuously detected in the web logs of a honeypot Apache server that was created prior to RAMPART implementation. The attacks generally consisted of a brute force directory traversal scan that specifically attempted to access the setup.php file in the phpMyAdmin, that many times originated from an anonymous proxy. A brute force attack is trial-and-error method to gather information it typically functions by making requests to a server using a set of predetermined values crafted by an attacker, and then analyzing the server's response [18], [19]. A directory traversal is an Hyper Text Transfer Protocol (HTTP) exploit that permits the attacker to access restricted directories, execute command, and view content external to the primary web server's root directory where application data is located [20]. The phpMyAdmin attack is a combination of these attacks and represents a creditable attack from the cyber threat domain.

2. Information Technology and Communications Services (ITACS)

ITACS incorporates and executes the most current cyber defense practices, policies, and technology available to monitor and protect the NPS networks. Using a defense-in-depth approach, they have incorporated firewalls, anti-virus, IDS, IPS, and application-based defensive components to protect the system. ITACS also conducts routine log analysis to detect network attacks. They rely heavily on log analysis to detect phpMyAdmin attacks making it an applicable use case for our project.

The operators at the NPS ITACS walked through five major phases in handling incidents. The five phases are identification, containment, neutralization, recovery, and assessment and documentation. Starting with the identification phase, operators identify the source, type, and effect of the attack on the affected system(s). They also examine symptoms of the attack and what services or servers the attack is affecting. Then, during the containment phase, operators take measures to contain the attack to prevent its spread to other systems on the network. Operators isolate the system in a localized way.

Next, during the neutralization phase, operators wipe and reimage government-owned client machines to neutralize and stop the attack. Since personally owned devices are not within the jurisdiction or purview of ITACS personnel, they provide users recommendations to remediate their machines e.g, wiping and re-imaging. Furthermore, the ITACS staff neutralizes and remediates servers coordinating with the system administrator. Afterwards, during the recovery phase, operators recover systems and return them to operation as soon as possible. Next, ITACS personnel methodically restore the systems from backups. Also, prior to returning the system to operation, they remove any license-limited tools and then systematically reconnect devices to the network. If a user on an NPS system receives a virus, the user's domain profile is cleaned. Next, ITACS re-enables the user’s domain account and resets their password. Finally, ITACS re-enables remote connectivity via VPN, wireless, or dial-in.

Last, during the assessment and documentation phase, operators thoroughly document the vulnerabilities that allowed the attack to be successful. By assessing and understanding the anatomy of the attack, layers of defenses and security procedures, ITACS personnel increase, modify as necessary, and enhance TTPs to prevent or mitigate future attacks. They conduct documentation during the course of the incident to capture coarse details. After the incident response, they conduct an incident debrief to corroborate and identify further details for the record. Then they review system configuration for vulnerabilities that permitted the attack to occur; they correct these vulnerabilities and deficiencies. Next, they update TTPs prevent similar future vulnerabilities and attacks. An IDS rule or rules are created to monitor and alert operators to specific network activity, particularly activity that signals an attack. An IDS rule is a specific network administrator defined criteria that when met could indicate an intrusion. NPS ITACS references a SOP to quickly step through the five phases; see Appendix D

III. MATERIALS AND METHODS

A. PROJECT APPROACH

In order to mitigate the manpower-intensive burden of log analysis, the project team attempted to automate the repeated actions found in the human work of log analysis. Time constraints were the main driver for the selection of methodology. Team RAMPART had six months to implement a network resident information apparatus capable of standards-agnostic incorporation of information. The approach taken was to build a VRDM executable information apparatus with a weighted scoring system to characterize anomalous network behavior either as an attack or normal behavior. We created a honeypot web server to obtain attack data and found that attackers generally scanned the namespace of the server from an obfuscated IP address and attempted to invoke PHP Hypertext Preprocessor (PHP) scripts in the phpMyAdmin program directory.

We postulated that a component, semantic-based information apparatus utilizing a weighted scoring system may determine if a specified risk level necessitated an automated response. This model operated under the premise that single instances of network behavior are sometimes insufficient to classify the behavior, and that some events can be more suspect than others. An automated analysis of the threat level of a phpMyAdmin attack would demonstrate the viability of using a VRDM executable information model as an automated analysis tool. This proof-of-concept could then be applied to sophisticated attacks with larger decision sets requiring an advanced criterion to be defined and met.

B. IMPLEMENTATION OVERVIEW

RAMPART is an implementation of a suspicion based intrusion detection information model created to demonstrate the concept of an intrusion detection and response model for network defense SoS [1]; see Appendix E. The component objects of the RAMPART information model are comprised of the following: the GINA kernel, the model specification, the specified data objects, the sources of the specified data objects, and the executable software identified to be part of the system process.

The information model implementation utilized distributed hardware, servers and software, and data elements, e.g., The Onion Router (Tor) watch list. We configured the Apache Web server to ingest data and process it in a manner similar to how a network administrator or an operator would do so to determine if an attack occurred. The model pulls database information, e.g., remote IP addresses, Uniform Resource Locators (URL), or page status from the Microsoft (MS) Structured Query Language (SQL) server containing the known Tor exit nodes and the MySQL server containing the Apache weblogs, and checks them against a criteria consisting of three indicators: 1.) a known Tor exit node, 2.) "phpMyAdmin" in the URL request, and 3.) "404" page not found status error. If two of the three indicators are met, then the model executes a response to block the possible attack through the Responder; see Figures 2 and 3.

The phases of the information apparatus specification consist of ingesting data, scanning web logs for patterns that are suspicious, correlating the patterns relative to IP address, summing the suspect pattern matches, and generating a response when a threshold value is reached. The conceptual model is depicted in Figure 2 and Figure 3. RAMPART employs two unique tables that act as a repository of tasks during continuous compare operations. The first table is a state table that maintains the unique identity of a web connection. The second table is an accumulator table where each suspect pattern is recorded as an identifiable attribute. Once the accumulator has an IP address record that has more than one suspect attribute, a special VRDM Form is invoked by a timed service procedure. The VRDM Form reads in the IP address of the suspect identity and forwards the IP address as a command line argument to a batch file. The batch file executes a Plink command to send Secure Shell (SSH) commands to the router to block the IP address.

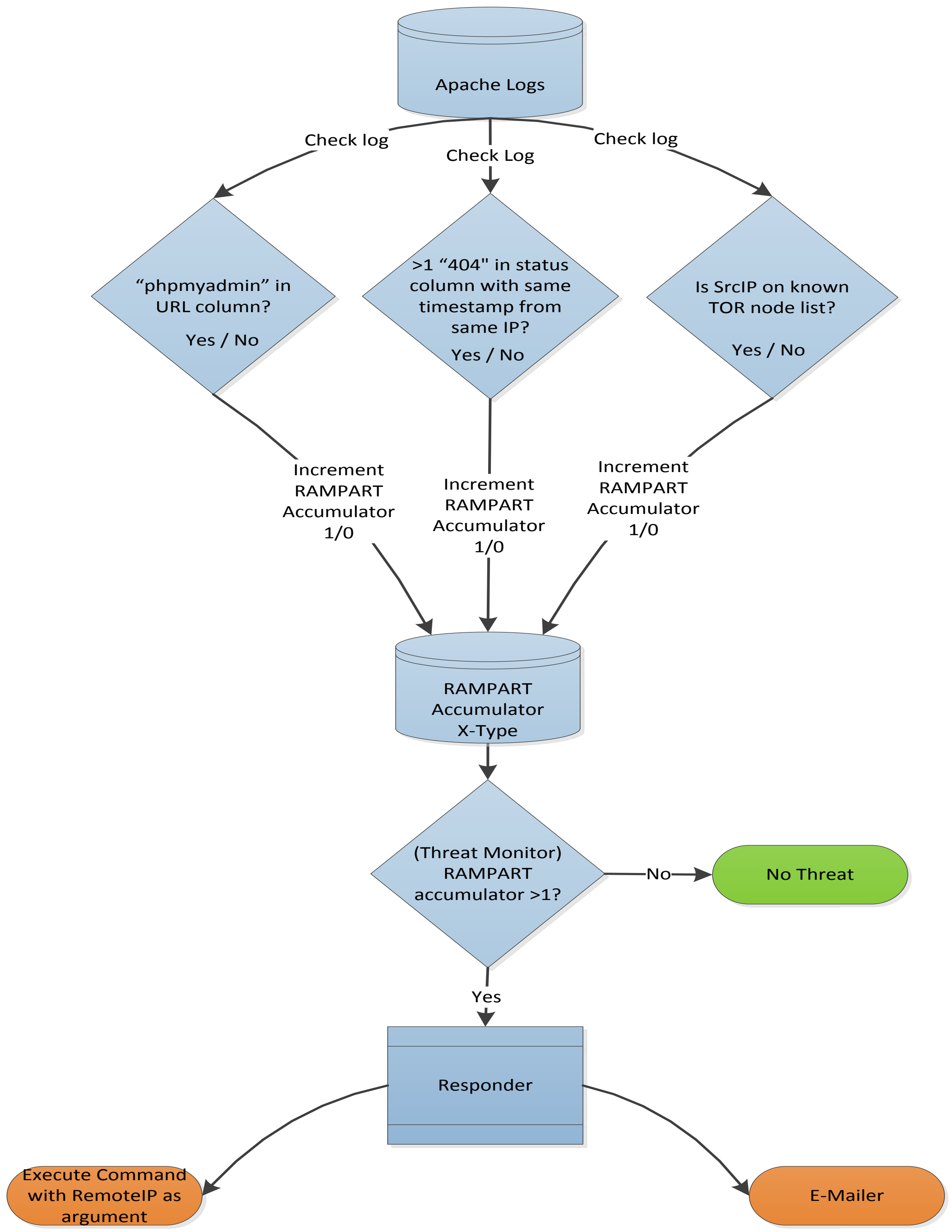

Apache Logs
Check log
Check Log
Check log
"phpmyadmin" in URL column?
Yes / No
>1 "404" in status column with same timestamp from same IP?
Yes / No
Is SrcIP on known TOR node list?
Yes / No
Increment RAMPART Accumulator 1/0
Increment RAMPART Accumulator 1/0
Increment RAMPART Accumulator 1/0
RAMPART Accumulator X-Type
(Threat Monitor) RAMPART accumulator >1?
No
No Threat
Yes
Responder
Execute Command with RemoteIP as argument
E-Mailer

Figure 2. RAMPART Conceptual Model. Each diamond is a VRDM decision that occurs in a parallel process and populates a special table (RAMPART Accumulator) with a binary result of each decision. The threat monitor sums the decision results and triggers the responder when any row decision sum is greater than one. The responder uses the IP address element to identify threats in a command line argument or email.

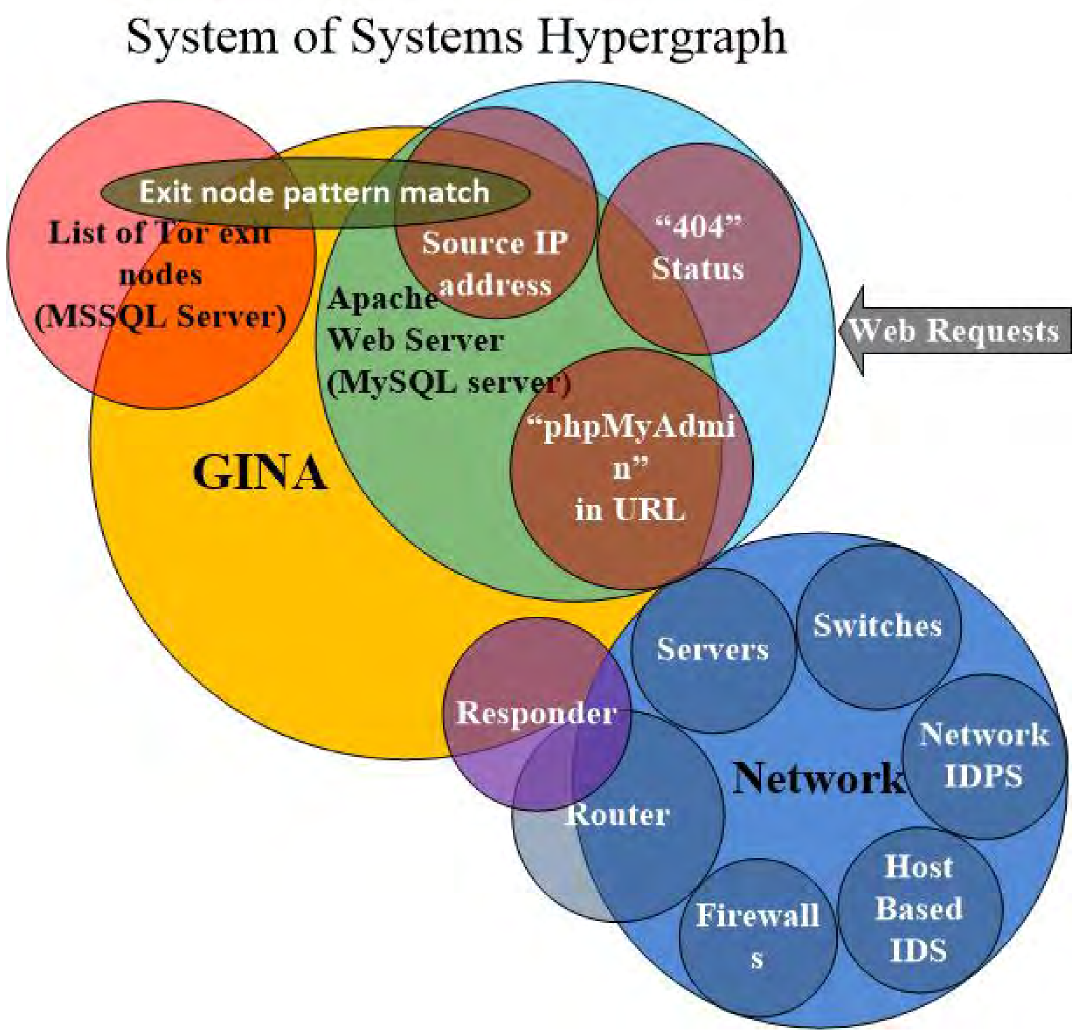


Figure 3. The SoS hypergraph shows the system components, data objects and network that interact with the GINA environment and are available to the information apparatus. The "Exit node pattern match" ellipse represents the portion of the VRDM information apparatus that specifies the interaction between a Microsoft Windows server and a Linux MySQL server.

C. RAMPART INFORMATION MODEL DECISION CRITERIA

The capstone project addresses three signatures indicative of our selected cyber attack use case: The phpMyAdmin attack.

- "404" page not found status error in the web log database.
- Known The Onion Router (Tor) exit node.

- “phpMyAdmin” is in the URL request.

Any one of these alerts may not be indicative of an attack. However, blocking of an IP based on a single alert criterion is impractical, as it would lead rapidly to a Denial of Service (DoS) to legitimate users. Combinations of any two of the indicators provide a more robust trigger for identifying malicious phpMyAdmin attacks from genuine users. RAMPART requires two out of the three indicators to be met in order to classify as an attack and warrant a response.

“404” page not found status error

The decision-making criteria for declaring an IP as a source of a malicious attack is based on input from ITACS operations and refined through server log analysis. From the log analysis, we determined that two “404” page not found status errors from the same IP address that occur within a two second window was sufficient to infer that malicious scanning was in progress. The “404” page not found status errors alone would yield too many false positive malicious activity, since the error can occur as a result of an unintentional activity, e.g., a typing error by the user.

Known Tor Nodes

RAMPART maintains a list of all known Tor exit nodes. It compares the embedded Tor exit node list against the source IP of the web request. If a match is found, the information apparatus increments the accumulator by one. Blocking all Tor users would prevent the legitimate users as well as the potential attackers. RAMPART must have a means to distinguish legitimate users from the attackers.

phpMyAdmin

A typical phpMyAdmin attack includes “phpMyAdmin” or similar syntax in the URL of the web request. However, legitimate use of “phpMyAdmin” may also occur especially by network administrators throughout their daily tasks. Consequently, you do not want to block all web requests using “phpMyAdmin” in the URL but be able to characterize the legitimate users from the attackers. Combining this indication with one of the other indicators can reveal a user’s intentions, i.e., hostile or not.

D. TRAINING REQUIREMENTS

Implementation of the model required training on key concepts of database management, Linux configuration, Apache web server configuration, router configuration, network hardware, and server hardware configuration prior to integration with VRDM. The RAMPART model implementation required training on the GINA framework for the capstone project team from Big Kahuna Technologies, the owner of the software patent for GINA. Training consisted both of on-site UI usage, and distance tech support for model configuration and implementation; approximately ten one-on-one sessions each five hours in duration. The last two training sessions configured the RAMPART model in less than ten hours. No programming was required with the exception of a batch script file that relayed a blocked IP address command to the router.

E. PHYSICAL SYSTEM BUILD

1. Original System Specifications and Changes

In the design phase of the RAMPART prototype, the team planned for physical and configuration challenges to ensure our build would be architecturally scalable, flexible, and compatible across disparate platform systems. The design began with ten Dell 1950 PowerEdge blade servers with dual quad core 2.66 gigahertz (GHz) processors and 16 gigabytes (GB) of Random Access Memory (RAM) that was upgraded from the original 8 GB. We also installed two 300 GB hard drives in each system with Redundant Array of Independent Disks (RAID) 0 configurations in each of the database servers. Each system came with two gigabit (Gb) Network Interface Controllers (NIC) built-in. One additional NIC was added to three of the servers to allow for a connection to multiple networks and the ability to separate and localize Internet traffic from the internal network traffic. In effect, this minimizes the logging of internal traffic.

2. System Repairs

Each of the servers had multiple hardware issues, e.g., inadequate cooling, hard drive controller cards with blown capacitors, or damaged cabling. Once a given problem was corrected, other problems arose that required corrective action. Power management was another issue, as power in the office was not sufficient to run all of the servers simultaneously. These power issues hindered the team's ability to keep all systems at full capacity. The power issue was resolved by transferring the servers to an ideal lab location. Once all hardware and configuration issues were corrected, the servers were left powered on to ensure uninterrupted and stable operation.

3. Hardware Layout

Switch (Brocade and Cisco)

Original configuration utilized a 24 port Brocade switch that was configured with three Virtual Local Area Networks (VLAN) for the Internet, internal network, and a demilitarized zone (DMZ). Upon setup and implementation, the switch validated connections with all computers connected to the network and the Internet, however it would not route any traffic between the systems or traffic out to the Internet. While troubleshooting the switch, the team eventually had to clear all the configuration settings and restart with a fresh installation. Rebuilding the switch and separating the networks again, yielded no change in the connectivity between the systems. Also while troubleshooting the switch; the team discovered a hardware problem that was beyond the teams' capability to repair. The team procured a 24-port Cisco switch from a point of contact at NPS and once installed, experienced no additional switching issues for the remainder of the system build.

IP Schema and Connectivity

The team used various Internet Protocol (IP) addresses and subnets to separate the networks into three subnets to minimize the chance of internal network traffic causing false positives during the testing phase, and to enhance security through the use of VLANS. The following local network IP addresses were used: the internal VLAN was

172.16.0.0/25, the Internet VLAN was 10.0.0.0/25, and the database servers were directly connected to the servers on 192.168.0.0/30 networks that the logs were originating from. The Internet was set up to route via the Ubuntu operating system (OS) using IPTables to control access and allow for the remote updating from the GINA service.

4. Operating System Selection

The team started with systems that were fully accredited for military use. The team experimented with different system configurations, including several different OS' to create an optimum working environment per our design.

Windows 2008 Server R2 Enterprise

The Windows 2008 servers were used in enterprise mode as both a database server and an Internet Information Services (IIS) server for the GINA framework. The team installed the Windows OS using default system settings. The installation of software required to establish communication channels between the nodes, e.g., the Open Database Connectivity (ODBC) connection is provided below under the software section.

Red Hat/Community Enterprise operating System

Red Hat 6.5 Enterprise was also selected since it is an accredited system. Initially, Red Hat was used to demonstrate the configurability of the information apparatus. The Apache server was initially implemented on Red Hat but required changing over to Community Enterprise Operating System (CentOS) because switching to CentOS 6.5 allowed for unrestricted installation of phpMyAdmin and all required dependencies with minimal errors. The second Linux server was installed with Red Hat and MySQL for the logging of the Apache web server access logs. Installation of MySQL and MySQL workbench for database administration was straightforward and required no additional external repositories.

5. Router and Trigger Mechanism

The trigger mechanism was an essential component of the RAMPART model. The trigger initiated a configuration change to the router that blocked or dropped sessions containing a phpMyAdmin attack. The team started with a Vyatta router OS supplied by NPS but modified the system to use Ubuntu with IPTables. The initial requirements for the trigger were the following:

- Single line of code executable on the command line interface (CLI).
- Command execution would cause a configuration change to the router.

The team encountered challenges creating and incorporating the trigger mechanism. The first challenge was that the GINA framework uses a Windows-based OS while the router used a Debian Linux based OS. The team decided to use the SSH protocol to remotely configure the router. SSH is a network protocol that connects two networked computers via a protected channel and enables secure data communication and other network services. Though the protocol was easily identified, a test bed was designed and implemented to test and develop the trigger. Two networked virtual machines (VM) were created to serve as the test bed for creating and testing the trigger. The next challenge was to find a SSH-capable program that could be executed on a Windows OS to SSH remotely into a Linux based OS.

Three options were possible. One option used a program called Plink, which is a CLI of the open-source terminal emulator, Putty. In a CLI, a user types commands to the program in consecutive lines of text. Putty supports numerous network protocols including SSH. It was originally created for Windows-based OSes [21]. Using Plink, we could create a SSH connection to the router. The second option was Cygwin. Cygwin creates a Unix-like environment and CLI on Windows OSes [22]. It integrates Windows applications to tools and data in its Unix-like environment. The third and final option was to use the protocol called "Expect." This surprisingly flexible and powerful tool allowed a command to be executed and the expected return to be anticipated and answered. Expect has a test terminal interface and automates interactive applications using common protocols such as telnet, File Transfer Protocol (FTP), SSH, and others [23].

Initial testing and research explored the capabilities and responses of Plink. Remotely logging into the router though SSH was relatively easy with Plink but executing configuration changes was difficult. Plink was designed for executing commands on another server and not a router. Cygwin did not readily have the command line execution required by GINA. The team's first success of executing configuration changes on the router was using Expect. Though verbose, Expect allowed for the unexpected responses from the router since one could program the anticipated responses. The difficulty was in creating a single command executable by GINA. The tested functional version of Expect was Cygwin-based, therefore the team returned to testing with Plink.

Adding to the difficulty of finding a remote SSH-capable command line program was the fact that Vyatta was bought by Brocade. Brocade removed the forums containing years of Vyatta user information. A new forum was started at Brocade's site, but the old forums never reappeared. Also, originally Vyatta was an open source router. After Brocade's purchase, it was changed to licensed software. The team had used a 60-day trial version of Vyatta but the trial period ran out during the creation of the hardware server test bed. This led to the team dropping its use and implementing an Ubuntu server using IPTables. This allowed the server to work in the role of a router, fulfilling our requirements.

Because team members had minimal experience in managing routers, a simple setup was desirable for updating the system and the IPTables to block harmful packets or requests. The team learned how to update and configure IPTables remotely. This allowed testing without compromising the servers' integrity. The team set up and configured the new router utilizing Ubuntu 14 Long Term Support (LTS) and the IPTables located in the /etc folder of the system.

The change to an Ubuntu server allowed the team to return to using Plink as originally intended. Plink gave the team a command line execution interface usable by the information apparatus as a Windows OS application. The VRDM information apparatus executes a batch file, running a Plink SSH command to remotely change the configuration of the IPTable routing on the Ubuntu server. This is the implementation of RAMPART's "trigger" mechanism.

Batch File

The Plink program allowed remote SSH login and command execution to update the router. The following lines were the result of the research and testing combined to create the batch file used by the information apparatus to execute the remote update to the Ubuntu router of the RAMPART system. The lines of code are as follows:

Line 1: `echo attempt %1 %date% %time% >> c:\temp\testresultkelly.txt`

Line 2: `%1. c:\temp\plink.exe -ssh -P 222 root@207.140.106.46 -pw rampart123!@# iptables -I FORWARD 1 -s %1 -j DROP`

Line 3: `echo success %1 %date% %time% >> c:\temp\testresultkelly.txt`

The first and third lines write the attacking IP address to a text file only for the purpose of generating a visual record of the IP address and the time of the attack for testing and evaluation of the system. The second line invokes the Plink command to SSH into the router remotely and updates the IPTables to drop the attacking IP address which is passed to the router through the variable.

6. Software Install

MS SQL (SQL Management Console)

The GINA framework operates primarily on Windows systems, though it is capable of communicating with Linux/UNIX. Utilizing MS SQL as the primary database we installed and configured the SQL Management console for easier administration of the various databases used in the information apparatus. Regarding ease of administration, the SQL Management console, once configured, facilitates remote administration of the databases that allows the network administrator to concentrate on other tasks.

Figure 4 shows the constituent databases and associated X-Types for the Apache web server logs, RAMPART VRDM model, and the responder X-Type used to invoke command line responses. The red ellipses in the figure illustrate Element comparison operations invoked by GINA Services. The X-Types are arranged in the logical progression of collection, detection, and response.

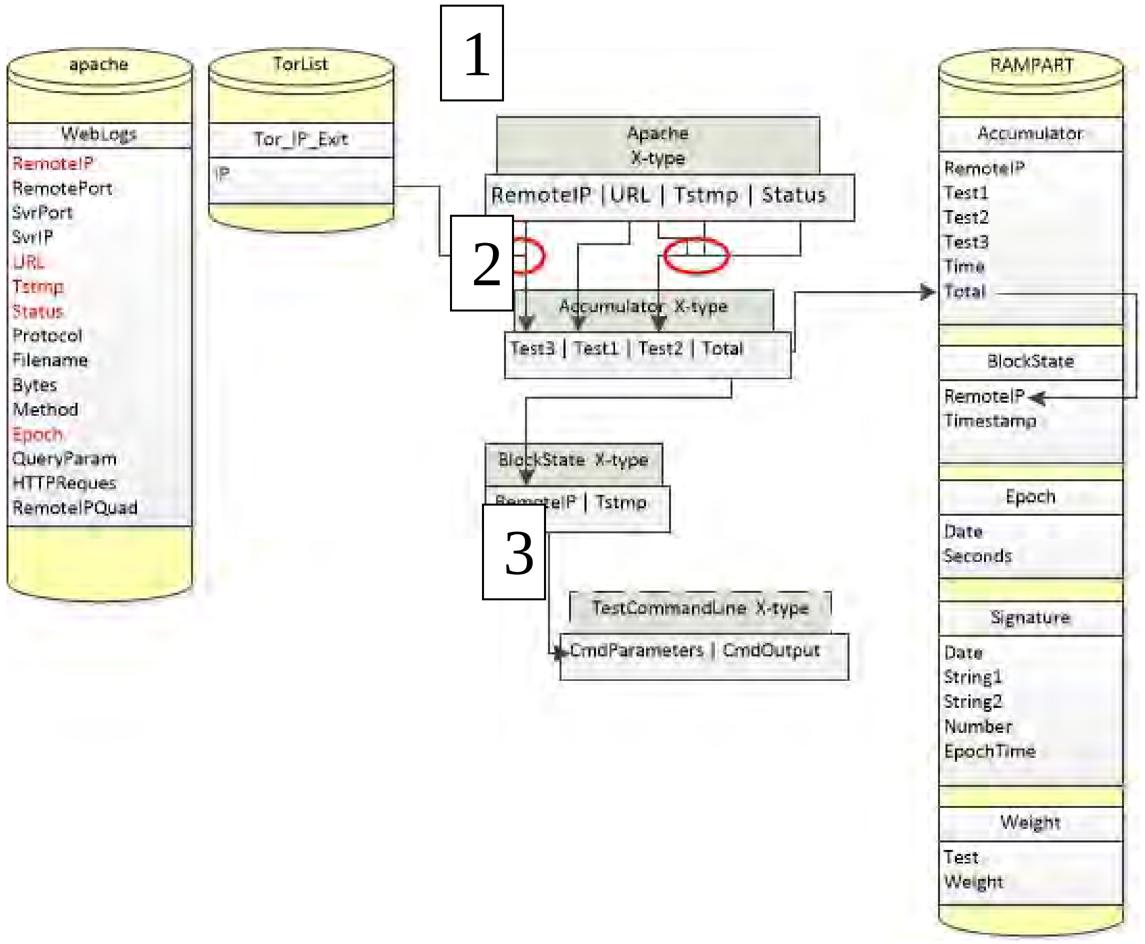


Figure 4. This figure shows the salient portions of the RAMPART databases and X-Types. The TestCommandLine X-Type has a dotted line around it because no physical source existed for the X-Type. The TestCommandLine X-Type only executed the *ExecuteCommand* service for each new Element. The red circles show where information is combined to make a deduction about the validity of a connection. Each database is shown with all tables and each table is shown with all columns.

Apache Server

After installation and setup of the Apache web server, we shifted the Apache logs from a standard file into a database. This allowed RAMPART access to the Apache logs; the logs are required to be in a database that is readable by the configurable information apparatus. Based on the information needed from the Apache logs the add-on module of mod_log_sql was initially chosen and implemented. As development of the model continued, we found that required information was not supported by mod_log_sql and another module was required. We decided to use the plugin module mod_log_dbd that was designed to use any of the log variables of Apache. The team configured the module. The httpd.conf file located in the /etc/httpd/conf directory must have the table name, column names, along with the username and password for the database with exact spelling and case or the connection will not be made and data will not be logged. Once the configuration was correct, data seamlessly flowed into the database. As testing began, the team also attempted to install and configure phpMyAdmin, the main attack vector of our testing.

PhpMyAdmin

PhpMyAdmin is available through various repositories and requires certain program dependencies. Red Hat requires a subscription and the ability to change the repositories you are subscribed to. Since the team did not have access to modify the Red Hat repositories, the system was switched to the more open CentOS 6.5. The exact repositories needed and the configuration are detailed in Appendix F.

MySQL (Workbench)

To allow ease of management of the web server logs in the database, a MySQL workbench was installed on both Linux and Windows servers. These allowed remote connectivity to the server for administrators.

ODBC / Visual C++

An ODBC plugin is required to convert a MySQL database to a MS SQL database. Since the GINA framework runs on MS SQL the ODBC plugin was needed to pull in the Apache Logs from MySQL. ODBC connections have specific prerequisite requirements to allow proper operation such as the Visual J# 2.0 Redistributable Package and the current Microsoft Data Access Components. Once installed and configured, a valid connection was made to the remote Apache database and the RAMPARTread in the Apache access logs. The logs were inputs to the RAMPART model.

IIS

The GINA framework is web based program for modeling database aggregation, decision-making and system behavior based. The configuration that controls the behavior is specified by the user in a web interface. The constituent components are Microsoft IIS server, MS SQL and DotNetNuke. At startup, the proprietary kernel creates a closed environment that may be accessed and managed through a web browser. The resulting VRDM environment is unique in that the programming actions are objects therefore; the entire semantically based information apparatus is configured [13].

System Updates and Linux Repositories

The servers initially were not updated automatically. However, after the test VM system was breached by an outside attacker, the physical system settings were changed to allow the automatic install of updates. The Red Hat and CentOS required specific programs and services dependencies to be installed before certain programs would function properly. Some of these can be installed during system installation but others require access to a particular repository before installation can occur. For a complete list of the required repositories; see Appendix F.

Remote Administration

Each Windows server had remote desktop enabled to allow remote administration. Each system was assigned a specific port and a rule was added to the IPTables in the router to forward the traffic for that port to the specific computer. This required changing the Windows configuration default listening port. The registry key of the Remote Desktop Connection (RDC) had to be modified. The path below is the exact registry key location:

"HKEY_LOCAL_MACHINE\System\CurrentControlSet\Control\TerminalServer\
WinStations\RDP-Tcp\PortNumber" [24].

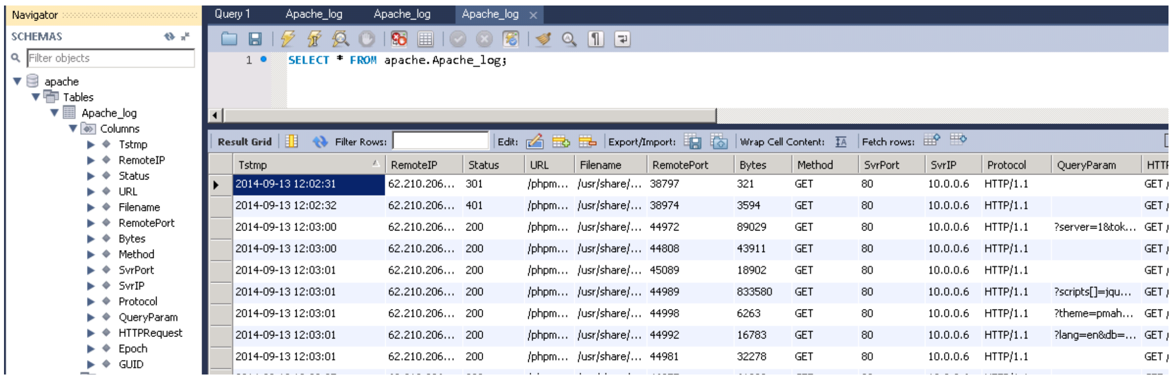
Each user must also be in the remote desktop users group before being allowed to connect from another machine.

Secure Shell

SSH was enabled and ports adjusted to allow multiple machines to have SSH connections through the router for remotely making configuration changes. Each machine's port was added to the IPTable of the router to forward any request on the specified SSH port, for example port 22, to the associated server for remote administration.

Model Components, Relations and Interactions

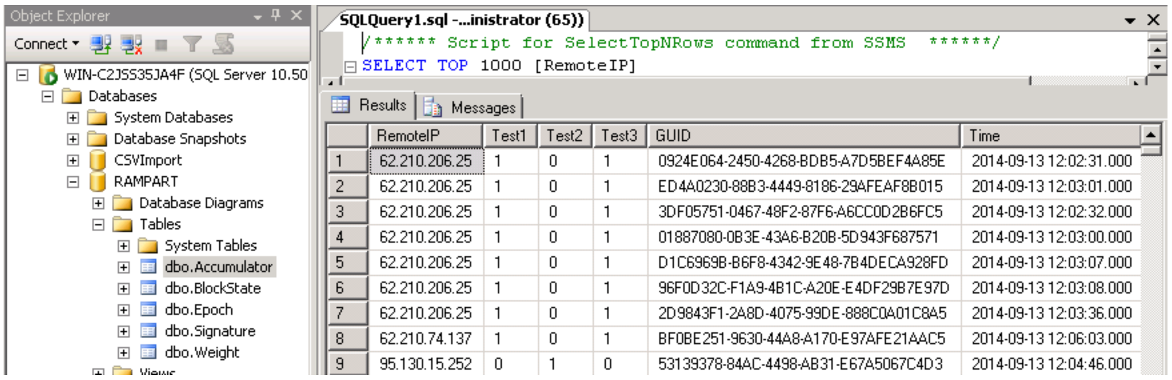
RAMPART used data from a Linux MySQL database and a MSSQL database; see Figures 5 and 6. Each column was integrated into the modeling environment as elements of six X-Types. These X-Types were Apache_log, Accumulator, BlockState, Epoch, Signatures, Weight, and TestCommandLine. All test data originated from the Apache_log X-Type and was manipulated according to values in the Signatures and Weight X-types. The TestCommandLine X-Type controlled the GINA service that executed a command whenever an Apache_log event failed a validation test.



```
SELECT * FROM apache.Apache_log;
```

Tstmp	RemoteIP	Status	URL	Filename	RemotePort	Bytes	Method	SvrPort	SvrIP	Protocol	QueryParam	HTT
2014-09-13 12:02:31	62.210.206…	301	/phpm…	/usr/share/…	38797	321	GET	80	10.0.0.6	HTTP/1.1		GET
2014-09-13 12:02:32	62.210.206…	401	/phpm…	/usr/share/…	38974	3594	GET	80	10.0.0.6	HTTP/1.1		GET
2014-09-13 12:03:00	62.210.206…	200	/phpm…	/usr/share/…	44972	89029	GET	80	10.0.0.6	HTTP/1.1	?server=1&tok…	GET
2014-09-13 12:03:00	62.210.206…	200	/phpm…	/usr/share/…	44808	43911	GET	80	10.0.0.6	HTTP/1.1		GET
2014-09-13 12:03:01	62.210.206…	200	/phpm…	/usr/share/…	45089	18902	GET	80	10.0.0.6	HTTP/1.1		GET
2014-09-13 12:03:01	62.210.206…	200	/phpm…	/usr/share/…	44989	833580	GET	80	10.0.0.6	HTTP/1.1	?scripts[]=jqu…	GET
2014-09-13 12:03:01	62.210.206…	200	/phpm…	/usr/share/…	44998	6263	GET	80	10.0.0.6	HTTP/1.1	?theme=pmah…	GET
2014-09-13 12:03:01	62.210.206…	200	/phpm…	/usr/share/…	44992	16783	GET	80	10.0.0.6	HTTP/1.1	?lang=en&db=…	GET
2014-09-13 12:03:01	62.210.206…	200	/phpm…	/usr/share/…	44981	32278	GET	80	10.0.0.6	HTTP/1.1		GET

Figure 5. This image shows a MySQL database on a Linux server. Records were populated from events on an Apache web server. There was a single table "apache_log" in the Apache database. Each event generated in the system log was parsed by "mod-log-dbd" and written to the database by "apr-dbd." The relevant columns were Tsp, RemoteIP, Status, URL, and Epoch.

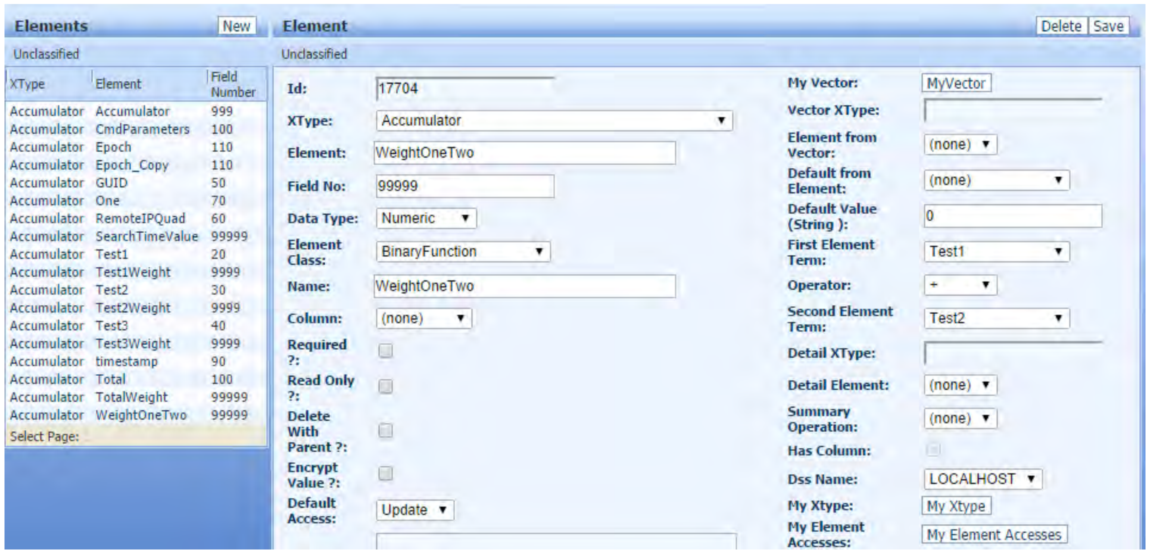


```
/****** Script for SelectTopNRows command from SSMS  ******/
SELECT TOP 1000 [RemoteIP]
```

	RemoteIP	Test1	Test2	Test3	GUID	Time
1	62.210.206.25	1	0	1	0924E064-2450-4268-BDB5-A7D5BEF4A85E	2014-09-13 12:02:31.000
2	62.210.206.25	1	0	1	ED4A0230-88B3-4449-8186-29AFEAF8B015	2014-09-13 12:03:01.000
3	62.210.206.25	1	0	1	3DF05751-0467-48F2-87F6-A6CC0D2B6FC5	2014-09-13 12:02:32.000
4	62.210.206.25	1	0	1	01887080-0B3E-43A6-B20B-5D943F687571	2014-09-13 12:03:00.000
5	62.210.206.25	1	0	1	D1C6969B-B6F8-4342-9E48-7B4DECA928FD	2014-09-13 12:03:07.000
6	62.210.206.25	1	0	1	96F0D32C-F1A9-4B1C-A20E-E4DF29B7E97D	2014-09-13 12:03:08.000
7	62.210.206.25	1	0	1	2D9843F1-2A8D-4075-99DE-888C0A01C8A5	2014-09-13 12:03:36.000
8	62.210.74.137	1	0	1	BF0BE251-9630-44A8-A170-E97AFE21AAC5	2014-09-13 12:06:03.000
9	95.130.15.252	0	1	0	53139378-84AC-4498-AB31-E67A5067C4D3	2014-09-13 12:04:46.000

Figure 6. This image shows the "central processing unit" of the RAMPART model. The RAMPART database exists on a Windows SQL server and has four tables: Accumulator, BlockState, Epoc, Signature, and Weight. Accumulator columns were RemoteIP, Test1, Test2, Test3 and Time. The values of each test were updated using VRDM transforms. The BlockState table maintained a list of each IP address that was blocked and when the block occurred. The Epoc table was a pivot necessary to convert the time stamps to Epoch time format. The Signature table held values indicative of a possible intrusion. The weight table assigned a severity value to each test.

The Accumulator is a special X-Type that serves as RAMPART's "memory" of the results of each multi-part test; see Figure 7. RAMPART validated a phpMyAdmin web request by testing additional characteristics of the request namely, obfuscation and scanning activity. A Transform updated the *Test* values when test conditions were met. Function Elements were used to sum the results of the *Test* values based on the weight of each test. In the phpMyAdmin use case, all weights were equal but larger models required variable test weights to better emulate the human decision-making process.

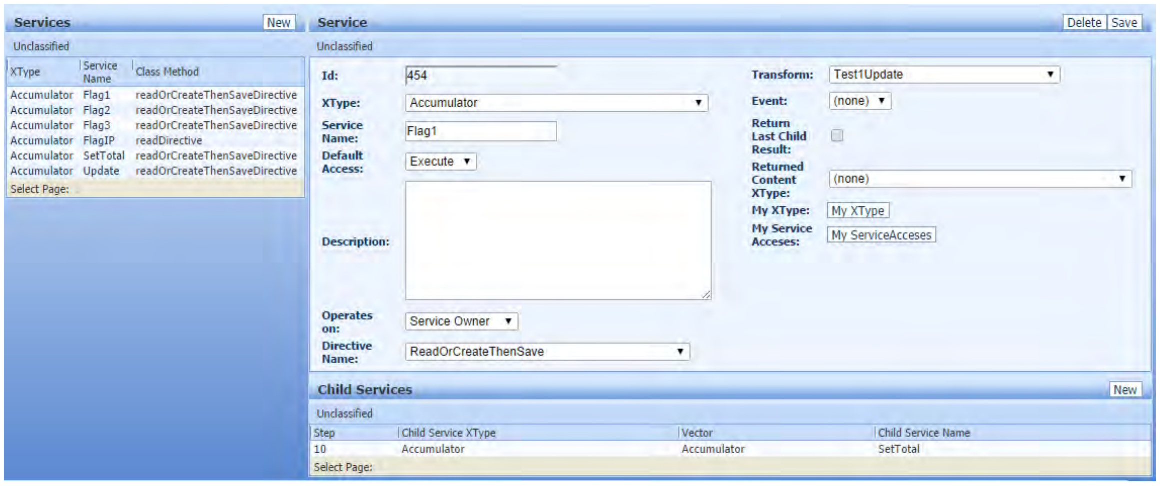


XType	Element	Field Number
Accumulator	Accumulator	999
Accumulator	CmdParameters	100
Accumulator	Epoch	110
Accumulator	Epoch_Copy	110
Accumulator	GUID	50
Accumulator	One	70
Accumulator	RemoteIPQuad	60
Accumulator	SearchTimeValue	99999
Accumulator	Test1	20
Accumulator	Test1Weight	9999
Accumulator	Test2	30
Accumulator	Test2Weight	9999
Accumulator	Test3	40
Accumulator	Test3Weight	9999
Accumulator	timestamp	90
Accumulator	Total	100
Accumulator	TotalWeight	99999
Accumulator	WeightOneTwo	99999

Id:	17704	My Vector:	MyVector
XType:	Accumulator	Vector XType:	
Element:	WeightOneTwo	Element from Vector:	(none)
Field No:	99999	Default from Element:	(none)
Data Type:	Numeric	Default Value (String):	0
Element Class:	BinaryFunction	First Element Term:	Test1
Name:	WeightOneTwo	Operator:	+
Column:	(none)	Second Element Term:	Test2
Required ?:	☐	Detail XType:	
Read Only ?:	☐	Detail Element:	(none)
Delete With Parent ?:	☐	Summary Operation:	(none)
		Has Column:	☐
Encrypt Value ?:	☐	Dss Name:	LOCALHOST
Default Access:	Update	My Xtype:	My Xtype
		My Element Accesses:	My Element Accesses

Figure 7. The image shows the Accumulator X-Type Element. The X-Type Elements consisted of Persistent Elements (field numbers that do not include 99), Vector Elements (field number 999), Reference Elements (field number 9999), and Function Elements (field number 99999). This Element, WeightOneTwo, was the summation of weights from *Test1* and *Test2*.

Services are the "workers" of the RAMPART model. They execute commands and other services when certain conditions were met. The services for the Accumulator X-Type were assigned Transforms in order to modify values in Elements; see Figure 8.



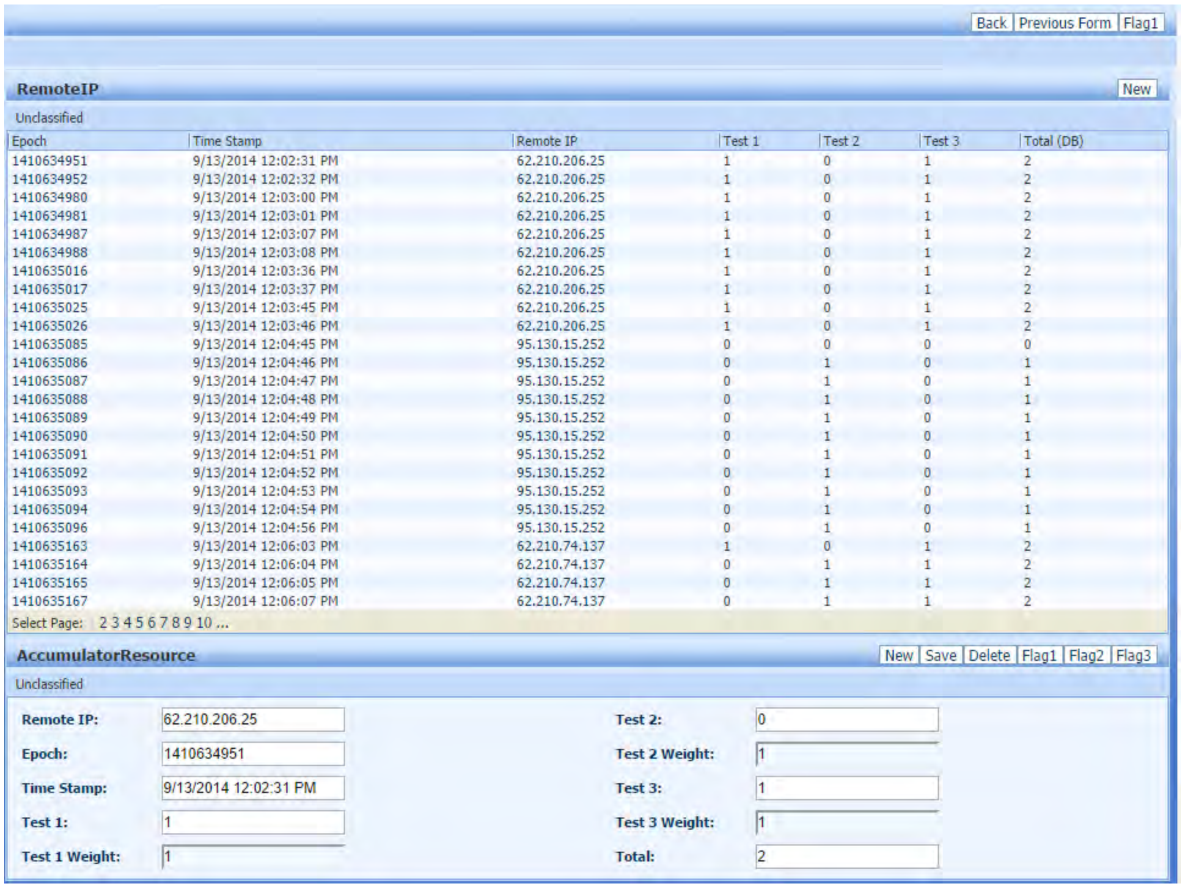
XType	Service Name	Class Method
Accumulator	Flag1	readOrCreateThenSaveDirective
Accumulator	Flag2	readOrCreateThenSaveDirective
Accumulator	Flag3	readOrCreateThenSaveDirective
Accumulator	FlagIP	readDirective
Accumulator	SetTotal	readOrCreateThenSaveDirective
Accumulator	Update	readOrCreateThenSaveDirective

Id:	454
XType:	Accumulator
Service Name:	Flag1
Default Access:	Execute
Description:	
Operates on:	Service Owner
Directive Name:	ReadOrCreateThenSave
Transform:	Test1Update
Event:	(none)
Return Last Child Result:	☐
Returned Content XType:	(none)
My XType:	My XType
My Service Acceses:	My ServiceAcceses

Step	Child Service XType	Vector	Child Service Name
10	Accumulator	Accumulator	SetTotal

Figure 8. The Accumulator X-Type services are shown at the top left of the image above. Each service was controlled by a transform that modified the value of the Flag Elements by using the calculation results from Function Elements.

Each test condition was assigned a service that "flagged" when Elements failed a validation test. RAMPART used *Test1* to determine if phpMyAdmin existed in the *URL* Element of the Apache X-Type and an affirmative executed the *Flag1* service. *Test2* determined if two or more "404" page not found existed in the *Status* Element, the *Timestamp* Elements were two or fewer seconds apart, and the *URL* Element values were not equal; if all were true, the *Flag2* Service was invoked. *Test3* determined if the *RemoteIP* Element matched any value in the *Tor_IP_Exit* X-Type Element *IP*; if a match occurred, the *Flag3* Service was invoked. Each of the Services generated results as they populated the Accumulator X-Type; see Figure 9.



RemoteIP

Unclassified

Epoch	Time Stamp	Remote IP	Test 1	Test 2	Test 3	Total (DB)
1410634951	9/13/2014 12:02:31 PM	62.210.206.25	1	0	1	2
1410634952	9/13/2014 12:02:32 PM	62.210.206.25	1	0	1	2
1410634980	9/13/2014 12:03:00 PM	62.210.206.25	1	0	1	2
1410634981	9/13/2014 12:03:01 PM	62.210.206.25	1	0	1	2
1410634987	9/13/2014 12:03:07 PM	62.210.206.25	1	0	1	2
1410634988	9/13/2014 12:03:08 PM	62.210.206.25	1	0	1	2
1410635016	9/13/2014 12:03:36 PM	62.210.206.25	1	0	1	2
1410635017	9/13/2014 12:03:37 PM	62.210.206.25	1	0	1	2
1410635025	9/13/2014 12:03:45 PM	62.210.206.25	1	0	1	2
1410635026	9/13/2014 12:03:46 PM	62.210.206.25	1	0	1	2
1410635085	9/13/2014 12:04:45 PM	95.130.15.252	0	0	0	0
1410635086	9/13/2014 12:04:46 PM	95.130.15.252	0	1	0	1
1410635087	9/13/2014 12:04:47 PM	95.130.15.252	0	1	0	1
1410635088	9/13/2014 12:04:48 PM	95.130.15.252	0	1	0	1
1410635089	9/13/2014 12:04:49 PM	95.130.15.252	0	1	0	1
1410635090	9/13/2014 12:04:50 PM	95.130.15.252	0	1	0	1
1410635091	9/13/2014 12:04:51 PM	95.130.15.252	0	1	0	1
1410635092	9/13/2014 12:04:52 PM	95.130.15.252	0	1	0	1
1410635093	9/13/2014 12:04:53 PM	95.130.15.252	0	1	0	1
1410635094	9/13/2014 12:04:54 PM	95.130.15.252	0	1	0	1
1410635096	9/13/2014 12:04:56 PM	95.130.15.252	0	1	0	1
1410635163	9/13/2014 12:06:03 PM	62.210.74.137	1	0	1	2
1410635164	9/13/2014 12:06:04 PM	62.210.74.137	0	1	1	2
1410635165	9/13/2014 12:06:05 PM	62.210.74.137	0	1	1	2
1410635167	9/13/2014 12:06:07 PM	62.210.74.137	0	1	1	2

Select Page: 2 3 4 5 6 7 8 9 10 ...

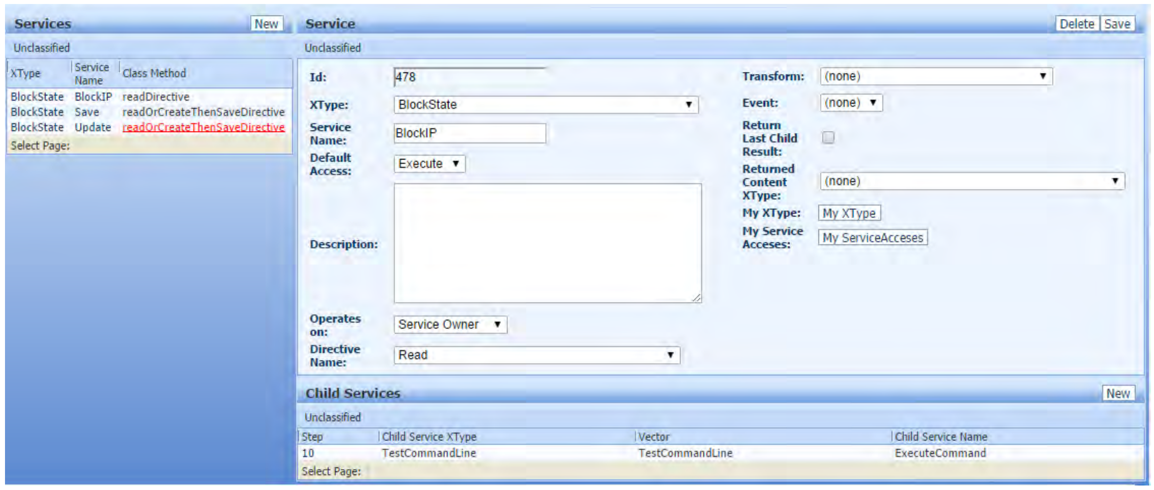
AccumulatorResource

Unclassified

Remote IP:	62.210.206.25	**Test 2:**	0
Epoch:	1410634951	**Test 2 Weight:**	1
Time Stamp:	9/13/2014 12:02:31 PM	**Test 3:**	1
Test 1:	1	**Test 3 Weight:**	1
Test 1 Weight:	1	**Total:**	2

Figure 9. This image shows the Accumulator records and the calculated total of the results of each test. Test totals greater than one indicated a threat. Test result totals of one or zero indicated no threat. This is an example of how a signature match can be evaluated under certain conditions to limit the number of false positives. The *Total* Element in the righost column contained summations of Test 1, Test 2, and Test 3. This sum was compared against a threshold value of 2.

If any *Total* Element value was equal to two or three, a Transform copied the value in the *RemoteIP* Element to the *Blockstate* X-Type. Each new Element in the *BlockState* X-Type called an ExecuteCommand Service; see Figure 10.



Services New

Unclassified

XType	Service Name	Class Method
BlockState	BlockIP	readDirective
BlockState	Save	readOrCreateThenSaveDirective
BlockState	Update	readOrCreateThenSaveDirective

Select Page:

Service Delete Save

Unclassified

Id:	478
XType:	BlockState
Service Name:	BlockIP
Default Access:	Execute
Description:	
Operates on:	Service Owner
Directive Name:	Read
Transform:	(none)
Event:	(none)
Return Last Child Result:	☐
Returned Content XType:	(none)
My XType:	My XType
My Service Acceses:	My ServiceAcceses

Child Services New

Unclassified

Step	Child Service XType	Vector	Child Service Name
10	TestCommandLine	TestCommandLine	ExecuteCommand

Select Page:

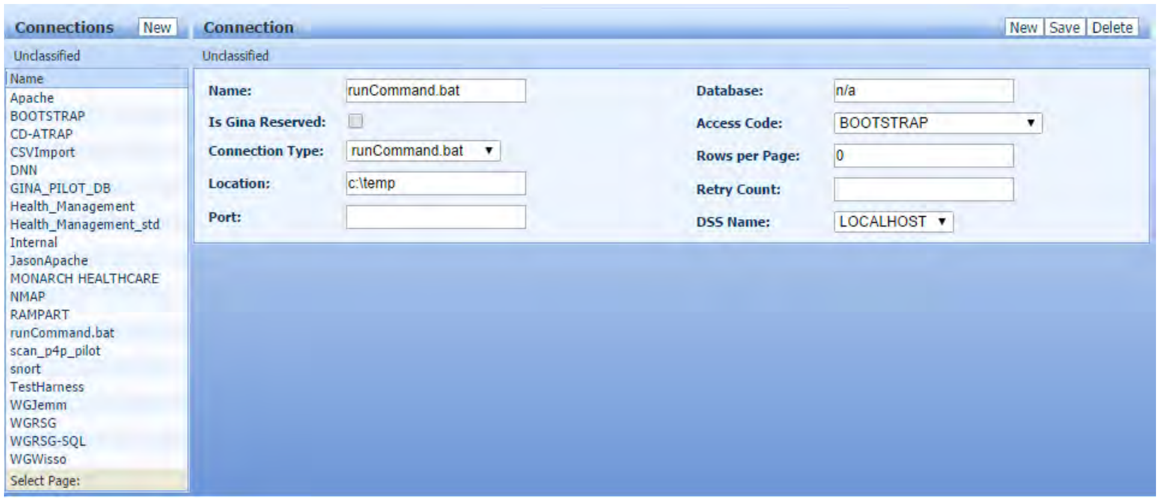
Figure 10. The Blockstate X-Type was appended to the RemoteIP Element of an Accumulator record that reached a row test sum of two. The ExecuteCommand Service was called for each new row in the Blockstate X-Type. The ExecuteCommand Service used the RemoteIP Element as a command line argument when each command was executed. Multiple arguments could be configured to be sent to a shell command.

The *ExecuteCommand* Service executed a shell command that was defined as a Connection; see Figure 11. The *RemoteIP* Element was the command line argument in this case. Additional command line argument Elements could be configured in the RAMPART model as necessary. The runCommand.bat file contained a simple two line SSH command invoked by the Plink SSH interpreter. The runCommand.bat file contained:

Line 1: `c:\temp\plink.exe -ssh -P 222 root@10.0.0.1 -pw rampart123!@# ./blockAndRemove.sh%1`

Line 2: `echo %1 %date% %time% >> c:\temp\Kellytest2.txt`

The script blockAndRemove.sh was located on the router device and served as a timer to unblock a connection after seven minutes. The second line outputted a log containing the time and IP address of each blocked connection.


Connections New

Unclassified

Name
Apache
BOOTSTRAP
CD-ATRAP
CSVImport
DNN
GINA_PILOT_DB
Health_Management
Health_Management_std
Internal
JasonApache
MONARCH HEALTHCARE
NMAP
RAMPART
runCommand.bat
scan_p4p_pilot
snort
TestHarness
WGJemm
WGRSG
WGRSG-SQL
WGWisso
Select Page:

Connection New Save Delete

Unclassified

Name:	runCommand.bat	Database:	n/a
Is Gina Reserved:	☐	Access Code:	BOOTSTRAP
Connection Type:	runCommand.bat	Rows per Page:	0
Location:	c:\temp	Retry Count:	
Port:		DSS Name:	LOCALHOST

Figure 11. The *ExecuteCommand* Service invoked the *runCommand.bat* Connection. The Connection shown is the file name and location of a shell command. Each different command was required to have a unique Connection.

RAMPART used a batch file that called Plink (an SSH interpreter) to send SSH commands to any SSH compatible device.

Working Model Implementation

The original design of the RAMPART system called for a nine-computer system that would detect various types of attacks. Additionally, the System used a decision tree with multiple tests to determine if the inbound packet was a threat from: a Tor type network exit node by the inbound IP address and/or the remote port of the request and verifying them against the know Tor exit nodes and ports commonly used. To complete these tasks the system was built using the nine computers connected through a switch. Each system was designed with a specific task in mind then the systems were networked together in three VLANS to separate the traffic looked at. Two of the other machines were set up as database machines to log traffic from the Apache web server and to run the GINA server. Figure 12 is a pictorial representation of the original system.

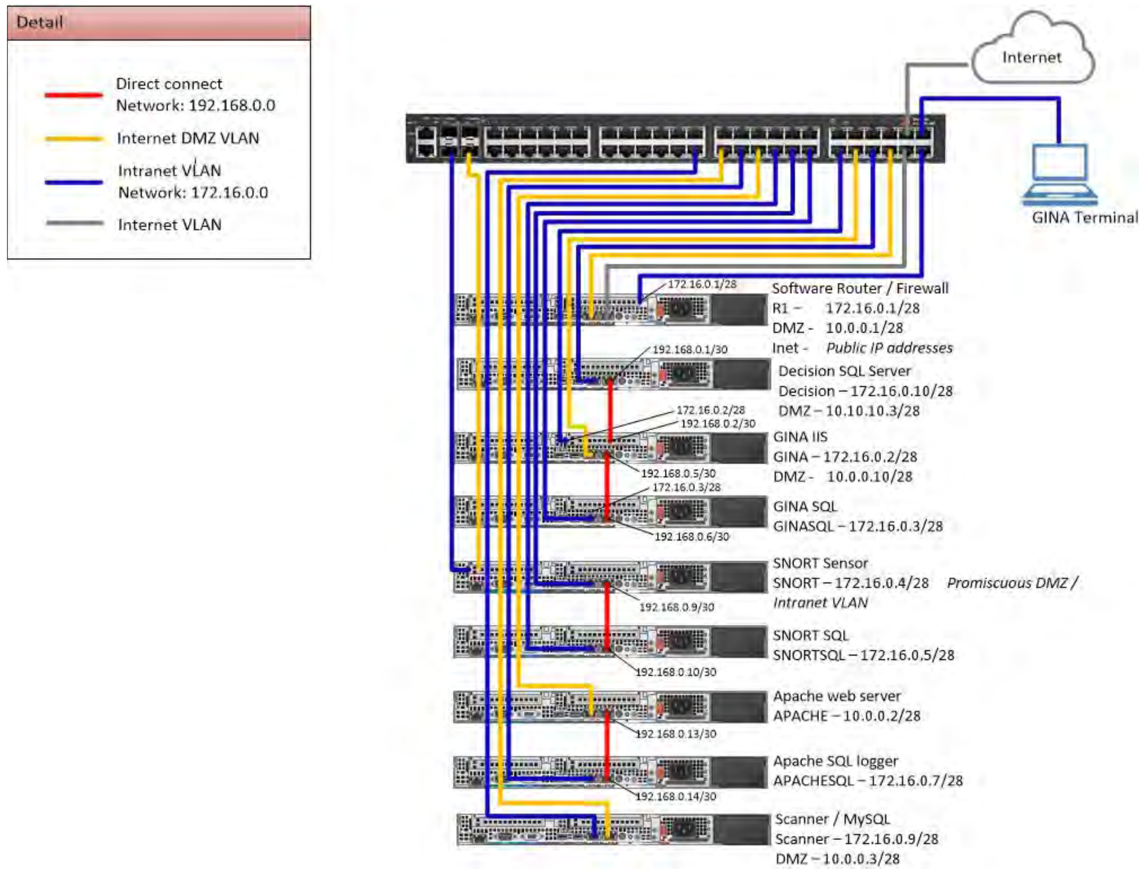


Figure 12. Conceptualized Hardware Configuration for RAMPART. The system was comprised of 9 servers separated into 3 VLANS to segregate network traffic. The red lines indicate the direct connection between databases.

The system was simplified due to time constraints and system issues. To simplify the model the team restructured the Tor detection section and removed the mapping section along with the Snort and strictly focused efforts on the phpMyAdmin attacks and the known Tor exit nodes. To accomplish this, we removed various sections of the model and lowered the required servers from nine to five computers plus the switch to allow for sectioning off the networks. The overall design of the system did not change but the team reduced the systems required for running specific portions of the information apparatus. The new VRDM information apparatus read Apache web logs from the MySQL database server and the GINA database servers and aggregated the information to run the tests against the criteria set forth in the information apparatus. The team created a prototype SoS out of five servers and a switch shown in Figure 13.

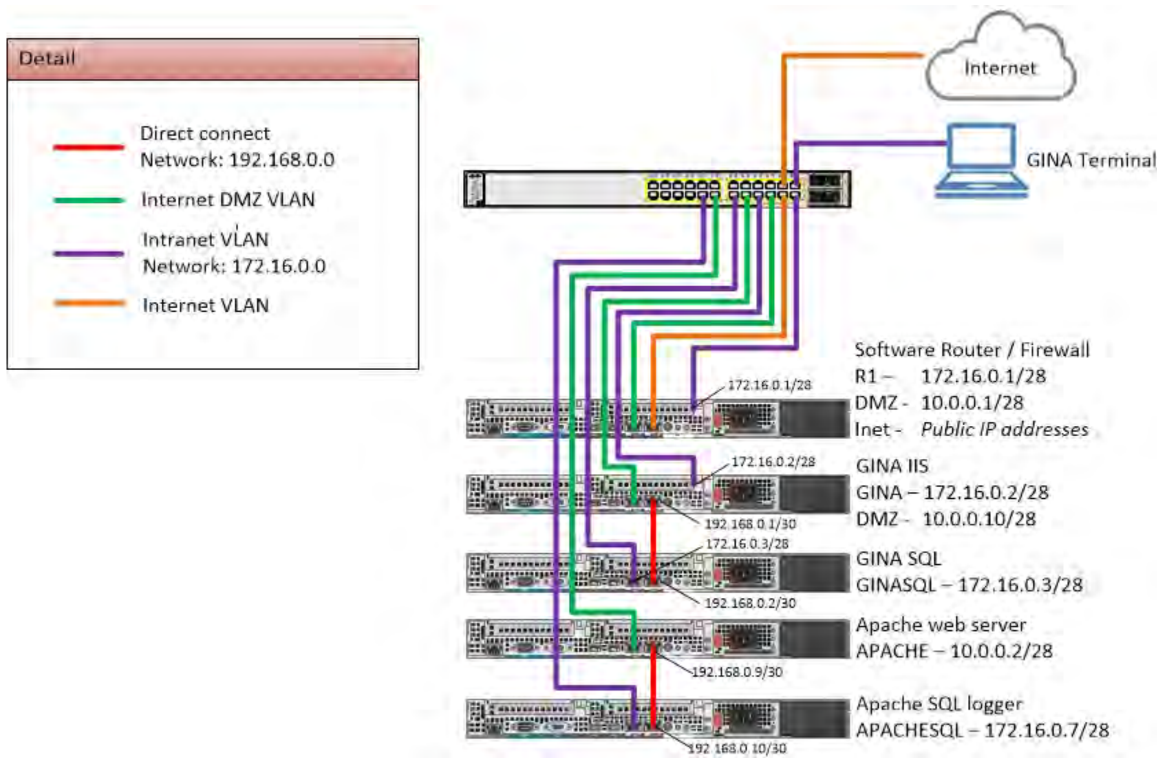


Figure 13. RAMPART hardware configuration. The final phpMyAdmin attack test bed was comprised of five systems, three VLANS and two connected databases.

THIS PAGE INTENTIONALLY LEFT BLANK

IV. TESTS AND RESULTS

A. TEST PLAN

RAMPART was designed as a system of systems that relies on the underlying functionality of three core components.

- MS SQL server 2008 backend database.
- Microsoft IIS running on Windows server 2008.
- DotNetNuke – an open source web content management system.

RAMPART used GINA as a database integration tool to integrate data sources from remote services. The capstone tested a Linux-based Apache web server with the following software implementations:

- MySQL as a Linux-based backend server for web applications.
- Mod-DBD as a parser of logs.
- APR_DBD driver as a database connector.
- PhpMyAdmin as a web based database administration tool.

Data pulled from a Microsoft IIS server represented the test network flow of information; see Figure 14.

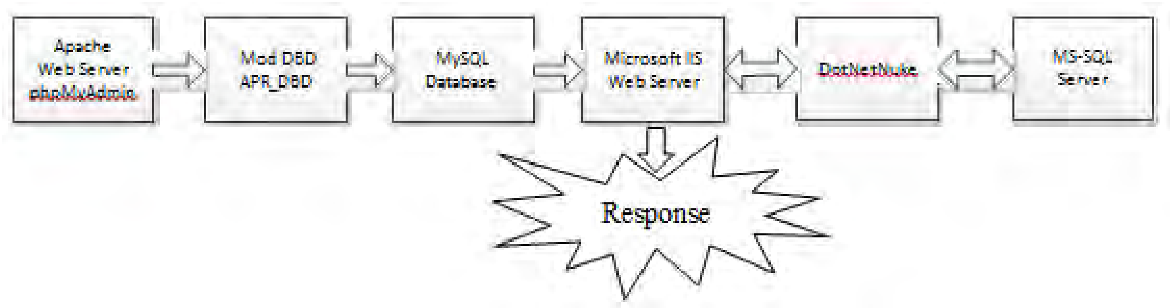


Figure 14. Above, the logical flow of data in RAMPART is shown. The test was designed to determine the integrity of modular information flow through RAMPART.

The project team exposed the Apache web server as a honeypot. Firewalls were disabled and the password was a common dictionary password. Possibility existed for malicious compromise.

1. Assumptions

- GINA could connect to a backend MS SQL database through proper configuration of DotNetNuke.
- GINA had access to Apache log database and table.
- Apache web server could log connection information to the backend MySQL database.
- Logging occurred in real-time.

2. Tested Features

(1) Database connection testing entailed the creation of VRDM sources mapped to columns in an Apache web log database. When complete and accurate data filled the GINA environment, the system had a successful connection.

(2) Performance testing entailed steadily increasing the amount of traffic to the web server until RAMPART became overwhelmed. Analysis of missed positives per total number of expected positives defined the hit rate at a given traffic level.

(3) Usability testing was to determine whether RAMPART analysis could be initiated without additional configuration and is completely automated. Recovery from a system failure should require nothing more than a system restart. The user interface must be capable of rendering decision data with relatively few clicks on a simple user interface.

(4) The percentage of true positives, false positives, and false negatives determined the accuracy of the system in a set of web requests. True positives occur when the system performed as expected. False positives occurred when RAMPART determines that a non-malicious activity is a threat. False negatives occurred when

RAMPART either missed a threat signature or did not detect all elements of threat triggers.

3. Approach

RAMPART testing began with a basic operability test of the user interface. Verification of RAMPART operation occurred before proceeding to tests of true positive detection and false positive detection. Team RAMPART determined the presence of the data and the accessibility of the data within a few clicks in a GUI. The test concluded with verification that a response occurred which resulted in a router update that blocked traffic from the IP address that related to the positive detection.

The test demonstrated the functionality of collection, detection, and response.

Collection

- We connected to the GINA user interface and determined if data from an Apache server was read into the framework.
- The data was visually confirmed to be accurate and complete.

Detection

- Connected to non-phpMyAdmin content from a Tor browser and scanned random URLs on the Apache server at the same time.
- Accessed phpMyAdmin using a Tor browser at the same time.
- Scanned phpMyAdmin URLs using a non-obfuscated browser.
- Accessed phpMyAdmin not using a Tor browser.
- Connected to non-phpMyAdmin content using a non-obfuscated browser.

Response

- Confirmed that the IPTable on the router updated to block the IP address of a threat.
- Time of response.

The system was configured to flag three events as suspect; see Table 1.

- Connection the phpMyAdmin site defined by any variant of "phpMyAdmin" string in the URL element.
- The presence of URL traversal scanning defined by at least two "404" page not found status elements within two seconds from the same IP address.
- A source IP address that matched an IP address on of a list of known Tor addresses.

Criteria	Grading Matrix: Two or more indicators constitutes a threat							
phpMyAdmin	X	X	X		X			
Scanning	X	X		X		X		
Obfuscation	X		X	X			X	
Test Outcome	Threat	Threat	Threat	Threat	No Threat	No Threat	No Threat	No Threat

Table 1. The RAMPART weighted flag truth table is shown above. Any two or more alerts indicated by an "X" result in a threat trigger. Each column indicates a possible test ($t_1 \ldots t_n$) and outcome is tallied at the bottom.

The test had two phases. The first phase determined if RAMPART worked and the second phase tested the limitations of RAMPART under increasing network traffic.

4. Pass/Fail Criteria

Test 1:

- Speed of detection must be within seven minutes of the suspected malicious event.
- Threat detection must result in an updated router IPTable.
- Results must have a 95% true positive hit rate and a 5% false negative hit rate.
- There must be less than 1% false positive hit rate.
- Data analysis of processed information can be accessed with no more than three clicks of a mouse in the RAMPART web interface.

Test 2:

Test 2 is similar to Test 1, but included a traffic analysis of events per second. It determined the number of events per second necessary to overwhelm RAMPART. Test 2 functioned as a stress test for RAMPART.

B. TESTS

This section describes a functional test of the usability of RAMPART and the functionality of a weighted decision model.

Testing supported the following objectives:

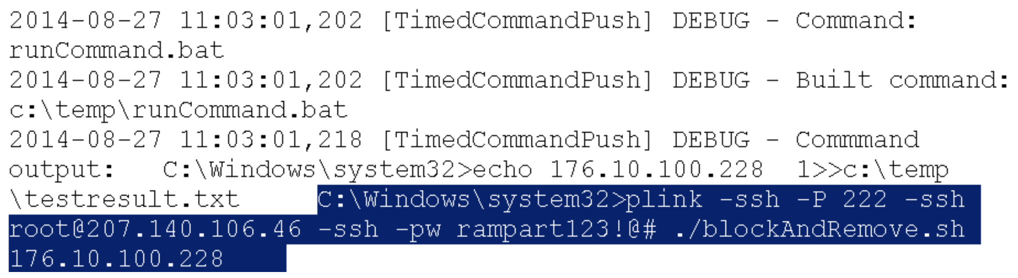
- Non-programming aspect of RAMPART as an Intrusion Detection System in a modeling environment.
- Effectiveness of RAMPART as an IDS model with active response.
- Effectiveness of RAMPART under a stress test.

C. RESULTS

Phase one testing took place over a period of ten days and consisted of running both automated and manual attacks from a Tor browser and a non-obfuscated system.

1. Test 1: Detection Rates

During phase one testing, the team encountered a few errors in the response of the RAMPART system. When confronted with a Tor based phpMyAdmin request, RAMPART flagged the request and attempted to block the IP address at the router. Though it was attempting to run the batch file, RAMPART did not invoke the command to block the IP. The team found that the GINA system was attempting to run the command from the folder C:\windows\system32 location which did not contain the correct executable file. The figure below is a portion of the GINA log file showing GINA attempting to run the command in the batch file; see Figure 15.



```
2014-08-27 11:03:01,202 [TimedCommandPush] DEBUG - Command:
runCommand.bat
2014-08-27 11:03:01,202 [TimedCommandPush] DEBUG - Built command:
c:\temp\runCommand.bat
2014-08-27 11:03:01,218 [TimedCommandPush] DEBUG - Commmand
output:   C:\Windows\system32>echo 176.10.100.228  1>>c:\temp
\testresult.txt     C:\Windows\system32>plink -ssh -P 222 -ssh
root@207.140.106.46 -ssh -pw rampart123!@# ./blockAndRemove.sh
176.10.100.228
```

Figure 15. Log file output of the GINA program calling the batch file to execute the update to the router

A modification of the batch file explicitly telling the program to run the file in C:\temp location corrected the issue and enabled RAMPART to correctly call the file. Then RAMPART blocked the IP address from the Tor network within seven minutes of the initial request for the phpMyAdmin page on the Apache server. When compared with the Tor exit node list embedded in the GINA database, results revealed a 95% detection rate with a 5% miss rate for the system. The system was also set up to detect and respond to a "404" page not found status error from the Apache server if the source originated from a Tor-based IP address. Again, RAMPART detected and blocked the threat within the required seven minute timeframe and updated the router to thwart the attacker. The third requirement of the system was to detect and respond to a scan of the Apache server for phpMyAdmin access from a non-Tor IP address. A scan such as this would generate "404" page not found status responses and meet the criteria: a phpMyAdmin in the URL combined with a “404” page not found status response. The team attempted to access the server with an incorrect phpMyAdmin URL generating a “404” page not found status error and elicited RAMPART to block the IP. The system responded within seven minutes, blocking the IP and added it to the FORWARD section of the router’s IP tables to drop any further requests from that IP address for the next four minutes.

The majority of the testing was completed from 28 August to 1 September. The last day RAMPART was flooded with over 37,000 events in about one hour, both valid and threat-based, to test the response time of the system and its capacity to handle large amounts of requests at once. Multiple VMs and a MACRO recording program were used to allow for automated testing of the system. Upon manual evaluation of the threat packets, we found that 2300 packets met threat criteria and invoked a response from RAMPART. After removing the duplicate IPs, 256 unique threats were isolated upon detection by the system. To resolve the exact response times, each initial request time was correlated to the first response time sent to the router for blocking. The delta between the times was calculated and then plotted on the graph below; see Figure 16.

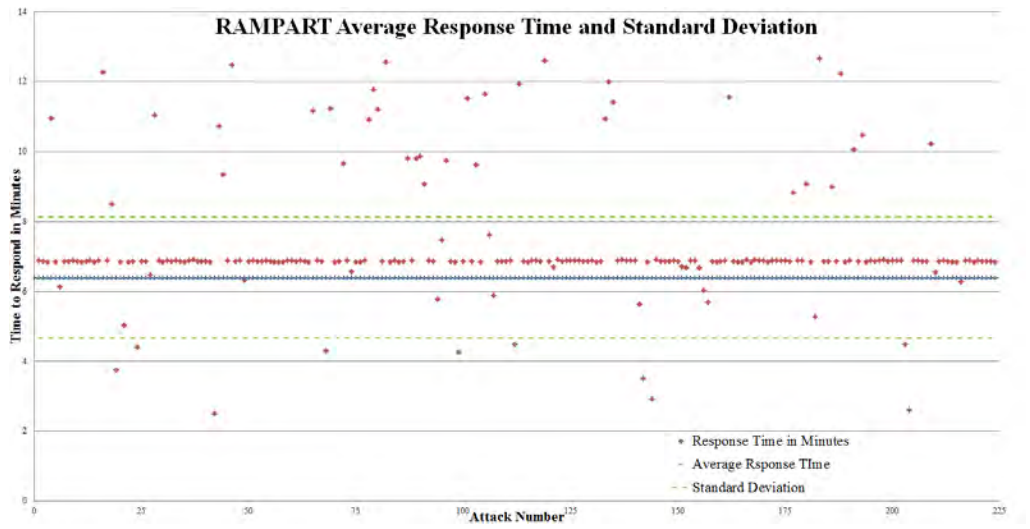


Figure 16. Plot of individual attacks numbered along the X axis with the time of response by RAMPART for the attack along the Y axis. The average RAMPART response time was just over six minutes. The average response time was calculated based on the total response time of all the attacks and the number of attacks for the given period of testing. The standard deviation was approximately 4.5 minutes.

The average response time for RAMPART was calculated based on the total response time of all the attacks and the number of attacks for the given period of testing. Overall, there were 256 threats processed by the system with a total response time of 1638.33 minutes for an average 6.39 minutes per event. Based on review of the data the high marks of 12 minutes were found to be IP addresses that just missed the previous database scan iteration of 6 minutes per model design, but were caught by the next scan and then blocked. The responses that were under the average were found to have been caught by the database scan just as it was finishing the first iteration. One way to draw the time closer together would be to increase the database scan rate from every six minutes to every three minutes for a full scan by limiting the amount of data that the system has to digest. Though the system did not achieve 100% detection on all attacks, RAMPART did detect and respond to most threats within seven minutes.

2. Test 1: Verification Tests

The team conducted several iterations of Test 1 to verify results. After clearing the table in the database we conducted additional test cycles, RAMPART was hit with over 1500 requests for various IP addresses consisting of 28 valid IP addresses. Legitimate, non-malicious IPs were set and expected to not trigger the system unless it faulted; conversely, the 134 attacking addresses were expected to be detected as intrusions and invoke responses. We found that, with a clear table, RAMPART successfully detected and responded to 100% of attacks without flaw nor latency; see Figure 17.

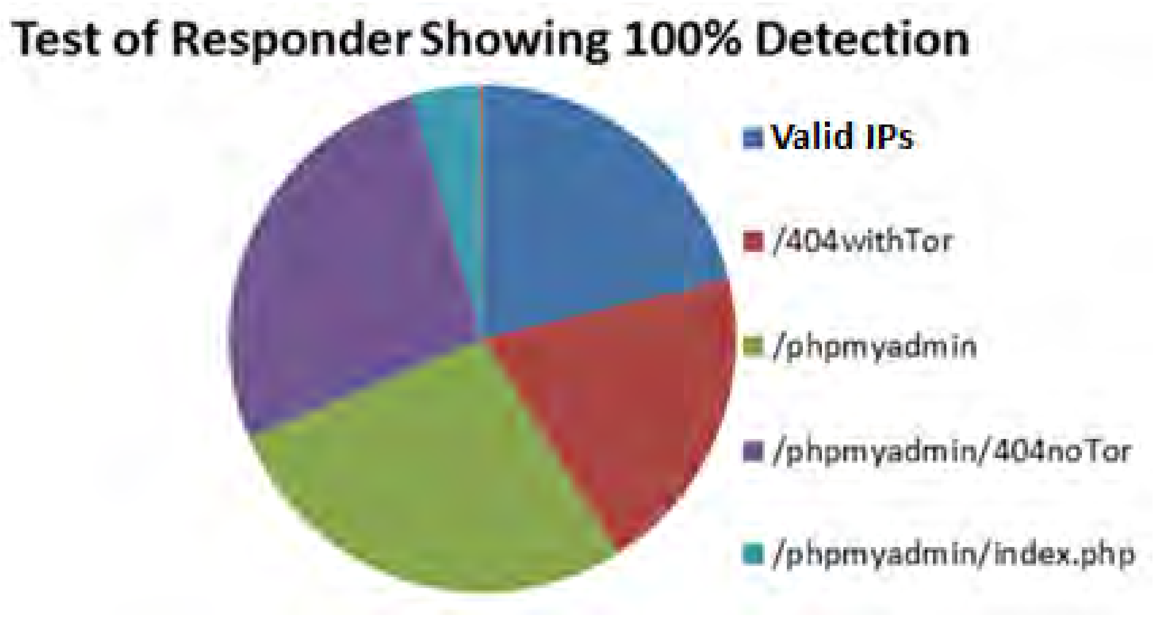
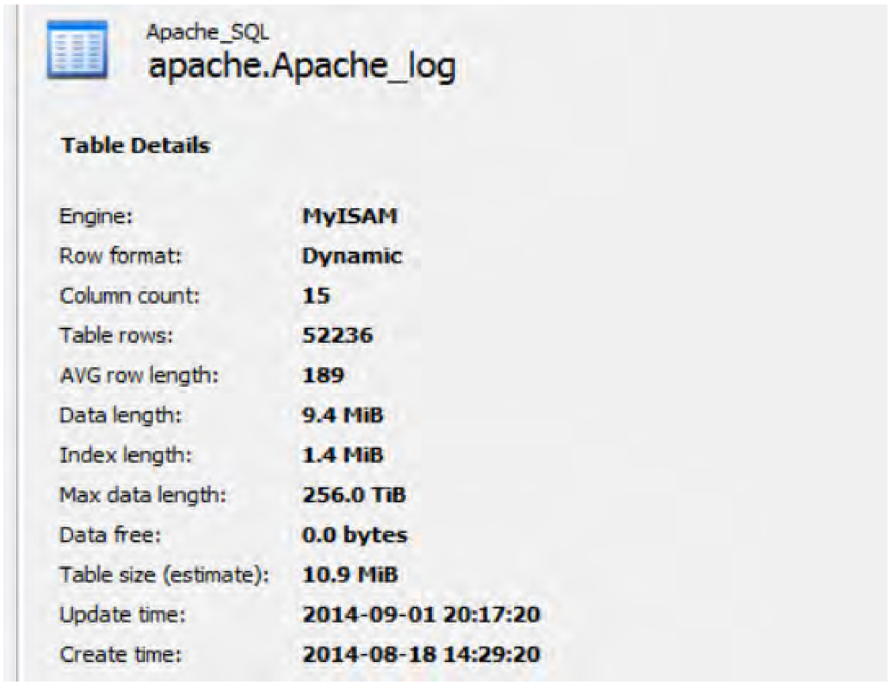


Figure 17. Chart depicting RAMPART's 100% successful detection and response to URL invoked requests. RAMPART permitted the 28 valid IP addresses to pass, and denied the 134 phpMyAdmin IPs.

3. Test 2: Stress Test

Team RAMPART conducted a system stress test of the RAMPART model utilizing multiple VMs with Easy Macro recorder installed to record and reproduce website hits in an organized fashion. A total of fifteen machines were built and setup to conduct the test. Six of the systems were set to connect through a Tor browser and automatically logged in and browsed the php database. This generated positive hits and triggered a response, while the rest just browsed the site as authorized users. Over four hours, these systems produced webpage requests and traffic to test RAMPART's ability to differentiate between an attacking IP address and a valid request. The system performed as expected. RAMPART updated the router to block the suspect IP addresses for a preset amount of time of four minutes. After approximately 49,000 to 50,000 rows in the Apache log database were logged, the information apparatus reached its processing limit and timed out in its search through the Apache web logs and allowed all traffic through; see Figure 18.



Apache_SQL
apache.Apache_log

Table Details

Engine:	MyISAM
Row format:	Dynamic
Column count:	15
Table rows:	52236
AVG row length:	189
Data length:	9.4 MiB
Index length:	1.4 MiB
Max data length:	256.0 TiB
Data free:	0.0 bytes
Table size (estimate):	10.9 MiB
Update time:	2014-09-01 20:17:20
Create time:	2014-08-18 14:29:20

Figure 18. Apache table information displaying the total number of rows in the table at approximately 52,000 entries or rows before the timeout occurs.

After the Apache web logs reached approximately 52,000 entries or rows, the queries would timeout on the database scans and allow all traffic through to the server. The fault lies in the information apparatus' ability to parse through all the logs on the Apache SQL server in the preset amount time of three minutes, one half of the full scan time, and find the new attacks. This inability to parse the log could lead to a DoS attack to flood the server with requests filling the log and causing it to not block valid threats entering the system. This would leave the system vulnerable to a more sophisticated attack after RAMPART falters. Though a typical system would not encounter a hit rate of 50,000 rows in the amount of time that RAMPART encountered them, the test provides a baseline for which to produce baseline criteria for future testing of the system.

V. DISCUSSION

A. SYSTEM PERFORMANCE

The system was designed to demonstrate that a weighted decision model is possible as a matter of configuration. RAMPART was the realization of the concept and created as a configuration. Common intrusion detection software such as Snort is known to provide automated response using software plugins such as SnortSAM and FWSnort. These programs use either an agent installed on the gateway device or intentionally flood the source or destination sockets with reset packets as methods of active response. Furthermore, Snort_Inline is a program that provides intrusion prevention functionality by converting a Snort server into an inline device. There are problems with each of these methods. Not all gateway devices capable of hosting agent software for SnortSAM, FWSnort reset packets can be blocked at the remote firewall, and Snort_Inline is vulnerable to a denial of service attack from programs like "Stick" that can send false positives for every address in the IPv4 address space. Cisco products such as Auto Secure and Adaptive Security Appliance are effective for Cisco products, but large distributed networks may not always have only Cisco products. The Federal Acquisitions Streamlining act provides for fair vendor competition and can induce a mix of proprietary products in large Federal networks that may not all be compatible with Cisco Auto Secure. RAMPART used configurable SSH commands to remotely reconfigure any SSH capable network device to overcome these limitations. There are also Snort preprocessors capable of creating decision trees that specifically perform the same multi-criteria decision tasks employed in the phpMyAdmin use case RAMPART model; however, programming specific preprocessors to perform this task is complex and the process is not conducive to the rapid reconfiguration needed for detecting new threats. RAMPART overcomes this limitation by introducing a decision as a modular object that can be reused and weighted as a matter of configuration that requires no programming.

Test 1 results demonstrated RAMPART's ability to detect and respond to phpMyAdmin attacks within an average time of seven minutes. As designed, results indicated a 95 percent true positive hit rate. The project's automated log analysis capability can reduce the time it takes organizations to detect cyber attacks from days or months to seven to twelve minutes. Decreasing the time required to detect an intrusion is a significant achievement that enhances an organizations' internal cyber attack detection capability. RAMPART's ability to provide an automatic response to halt attacks by reconfiguring the routing service also greatly improves an organization's ability to weather cyber attacks effectively. Even before a breach can occur, the automated response from RAMPART can stop the attack. Instead of days or even months containing an intrusion, the attack is halted before it can penetrate the network perimeter of the organization. This capability also has the extra benefit of avoiding the possibility of critical data being exploited and exfiltrated from the network despite early detection.

The second test, Test 2, stressed the system. The goal was to simulate heavy traffic use on the website to observe the ability of RAMPART to process all the events in a timely fashion. Despite the stress on the system, RAMPART updated the router IPTable, quickly adapting to the dynamic environment. The system performed as expected, updating the router to block the suspect IP addresses for a preset amount of time of four minutes.

B. LESSONS LEARNED

SME required

RAMPART integrated numerous expertise fields and computer applications into a single capstone project. In order to successfully implement the system, team members expanded their knowledge considerably in the following areas: Cyber Threat Domain, MS IIS, MS SQL, MySQL, Ubuntu, Apache, IDS/IPS, Vyatta, Plink, Cygwin, Expect, SIEM, current cybersecurity measures, VMs, Tor, GINA, Innovation, KM, and MCDA. This is not an all-encompassing list but adequately represents the wide breadth of knowledge needed to implement RAMPART. The team heavily relied on the technical expertise of several experts as well as NPS faculty and staff.

System Components

While the information apparatus is relatively simple to use when configured, each system component incorporated in the SoS must be itself configured and executed appropriately. The GINA framework can integrate many disparate components, but each needed to be configured correctly to communicate with GINA. A misconfiguration of any single component of the SoS can negatively impact the overall performance or even cause it to fail. Proper configuration of each component and expertise in its function can overcome many technical obstacles that arise in the system implementation.

Troubleshooting

Understanding where component system misconfiguration and the information apparatus implementation can malfunction presents trouble shooting challenges; however, these were remedied through reconfiguration of the information apparatus. Any SoS has numerous disparate components. Failures in the proper function of any of one of these components can cause a malfunction requiring troubleshooting. However, isolating the exact cause of the problem is difficult and requires expert technical knowledge and superior troubleshooting skills. Typically, the team found the best course of action to be a general reconfiguration of the entire system or following the information apparatus step by step until the problem was found; see Appendix G.

Configured vs Compiled

Many software programs are written with one configuration set and only allow small modifications to its base programming before requiring new programming code and recompiling. Adding another system to the original software would require a computer programmer to modify the underlying software code and then recompile it. However, a VRDM information apparatus would only require a configuration change. This saves a significant amount of time in comparison to traditional programming methods, and the configuration changes would not require a computer programmer to execute the modifications. Operators could configure RAMPART to incorporate numerous other cybersecurity components and aggregate their logs into the system. New security policies, system and network components, and new cyber threat vectors not previously addressed can all be integrated into the system in order to adapt it to the dynamic cyber threat environment. Through information modeling, RAMPART could then include these changes without requiring new programming or recompiling.

C. IMPLICATIONS

VRDM information apparatus' could have significant impact on cybersecurity. The near real-time capability to detect and respond to attacks requires the ability to rapidly parse the web log database against the set cybersecurity criteria. This requirement can overwhelm the VRDM information apparatus if it is not adequately equipped and configured. However, the system has demonstrated the ability to process approximately 50,000 lines of table rows before degrading. The database rows could be populated by web, system, or application logs giving network administrators extreme flexibility in their use of the system to protect the network. RAMPART may be a viable alternative or replacement to current cybersecurity mechanisms, approaches, and procedures. Though this capstone is the implementation of a proof-of-concept model, the envisioned system would be able to aggregate data from any system with the capability to store their logs in a database.

RAMPART shows high rates of true positive hits combined with a small percentage of positive false hits. This accuracy is a strength, though it cannot claim to be impregnable. The system thus allows operators to use automated log analysis. Thus, they can focus on other cybersecurity practices instead of spending critically valuable man-hours on time-consuming log analysis.

RAMPART could be an affordable alternative to many cybersecurity solutions. Automated log analysis reduces the time and personnel required to conduct effective, routine log analysis to protect the network. Also, the system is simple to use and manage, once it is configured. Personnel can be rapidly trained to proficiently use RAMPART.

Integrating multiple data sources into RAMPART will enhance situational awareness and allow operators to efficiently discover, remediate, and mitigate attacks, thereby improving critical decision-making.

THIS PAGE INTENTIONALLY LEFT BLANK

VI. CONCLUSION

Cyberspace is an inherently open and unsecure domain wherein cyber attacks threaten the network integrity of organizations and significantly influence the United States' overarching strategy on national security. Hence, cybersecurity is critically important to modern, connected societies, including large organizations, militaries, and governments. Many network intrusions remain undiscovered for months. Reducing the time required for detecting, containing, and mitigating cyber attacks can significantly increase the cyber defenses of any organization. By implementing innovative technologies, the DOD will be better-equipped to defend against cyber attacks. Furthermore, extensible system technologies, e.g., VRDM information modeling could be easily modified to stay ahead of evolving cyber threats, e.g., zero day exploits.

The team used VRDM to methodically build and configure a system to automatically aggregate disparate data and information into a single system. This process, in effect, resolved the human-intensive task of log analysis, specifically the detection and subsequent response to cyber attacks. The automation of this process exhibited a near real-time capability and solution to identify and defend against phpMyAdmin attacks. RAMPART's automatic response capability blocked many of these attacks before they could breach the network and cause damage.

RAMPART is a proof-of-concept demonstrating VRDM in cybersecurity for and network defense of a SoS. The capability of the GINA framework to aggregate data from multiple sources and perform connection validation of online identities using configurable, weighted decision models, makes it an ideal platform for connecting disparate cyber defense tools into a SoS. RAMPART does not require programming to expand and incorporate new cyber defense components. It automates log analysis providing operators a more efficient way to examine of network events, enabling them to detect and respond to cyber attacks quickly. The automatic response from RAMPART can potentially block many attacks before they exfiltrate crucial data or deliver an attack payload. RAMPART potentially could be the cornerstone of an organization’s cybersecurity portfolio.

A. FUTURE WORK

The working prototype has demonstrated the ability of a configured VRDM implementation to detect and respond to a threat using human decision-making criteria. Additional criteria can be added, thereby providing increased functionality to the base model. RAMPART can be used to explore other network defense applications.

Suggestions for extending RAMPART include:

Tor Scanning of Network Systems

The Tor network obfuscates a user’s IP address; however, RAMPART could easily monitor for obfuscated attacks through the use of Tor exit node detection, in conjunction with searching for calls for specific services not used via the Tor network.

Network Mapper Scan for Open Ports

Nmap is a useful, multi-purpose tool for administrators to monitor system information available to the public. Conversely, it is a powerfully exploitation tool to cyber attackers, who could perform scans of an organization’s network ports. Notably, they can scan for open ports on client servers and hosts that are accessible via the Internet. Using Nmap, the attacker can record all open ports and formulate a future attack. The ability for network administrators to recognize these external port scans and correlate them to other possible attack criteria on the network would provide a critical advantage in the defense of networks.

SQL Injection Attacks

SQL injection attacks can be some of the most devastating attacks. If the database server is misconfigured, an attacker could connect and gain remote access to the system. In some cases, the attacker could obtain administrative privileges to conduct further exploitation or cause serious damage. RAMPART could thwart these attacks through simple checks, such as looking for behaviors such as over use of a specific string in the URL. This could prove to be a pivotal step toward heightening network security.

Cyber-Defense Mechanisms – Anti-virus, IDS/IPS, Firewalls

Anti-virus companies play a vital role in providing network security. If anti-virus software were to be combined with the RAMPART response system, it could block viruses and monitor for possible data exfiltration or the use of malware. Also, IDS and IPS such as FireEye and Snort can detect intrusions, but they do not necessarily actively respond to the intrusion (in near real-time or at all). IDS and IPS detection capabilities could be coupled with the RAMPART response capabilities, making it a more capable cyber defensive and preventative tool.

Log Analysis

Many systems output security logs to databases, allowing them to be rapidly queried. RAMPART could take specific segments of a database as criteria in its decision process and allow system administrators to specify certain log attributes and possible flags. These log attributes and flags could alert the administrator to unexpected changes on the system or network, e.g., a “new user” added in the middle of the night when no one is expected to be logged into the system. Network administrators could tailor the system for a given OS on the network.

Dynamic Growth of IP Blacklists; Statefulness

RAMPART can communicate with any database through various network connections, thus the ability exists to centralize and update this list globally without network administrator action. This capability would streamline the process to update IPTables across the entire DOD enterprise, creating a more robust defensive posture.

Increase in the Database Parse Capability in the GINA Framework

Currently, the GINA framework searches for triggering criteria within the database every three minutes. If the system cannot make it through the full database within the three minutes, a timeout occurs. Any event that occurs after the timeout will not be scanned for the triggering criteria. An increase in the speed of the GINA architecture will allow for a more efficient style of information parsing which would increase effectiveness. Alternatively, an automated script could be developed that archives the database when it reaches a preset size. Either of these solutions would allow the system to continue to protect and respond to threats in a dynamic, automated fashion.

Outdated, Legacy Systems

RAMPART could be configured to incorporate functional and valuable legacy systems. The DOD continues to depend on numerous legacy systems, but replacing these systems across the entire military is exceedingly expensive in a fiscally restrained environment. In some cases, these systems lack adequate cybersecurity and struggle to run modern cyber defense applications. However, RAMPART could be used to support log management. This would not only improve their cyber protection, but may also continue to extend their usefulness and function.

Insider Threats

Insider threats are a national security problem and have recently risen as a security concern for many intelligence agencies. RAMPART can monitor for suspicious indicators within system databases and network log files. For example, network administrators could configure RAMPART to look for users who attempt to copy massive numbers of files to portable media such as optical disks or USB drives. The ability of RAMPART to manage multiple decision criteria could provide additional protection to an organization against insider threats.

As attackers develop and propagate new viruses and exploits in cyberspace, the threat to network systems is always evolving in complexity and scope. DOD cyber forces should be equipped with state of the art and innovative technologies and tools. All the aforementioned future work could significantly strengthen network security. They are also natural extensions of the work presented in this paper.

APPENDIX A: GLOBAL INFORMATION NETWORK ARCHITECTURE

Development on GINA was launched at NPS in 2004 as a Cooperative Research and Development Agreement. In 2005, GINA was a DOD Information Assurance Certification and Accreditation Process (DIACAP) Certification Class 3 Network-Aware Business Data Management System, and was released and integrated into various U.S. government organizations in 2006 [25].

GINA is an executable, component-based, platform agnostic, model-driven architecture that provides data, information, and knowledge interoperability, and enables the customization of services for systems. It is a modeling environment used to configure VRDM implementations into executable systems. Research scientist and developer of GINA DPIMS [26] Anderson explains, “GINA is able to take programs and devices and represent them in a way that is understood by every other represented application in the GINA environment. Configuring—not programming—creates relationships between the individual components so that stove-piped technologies become accessible and part of the greater GINA universe” [27]. Operators at NPS’ ITACS employ GINA to facilitate data sharing across a wide and diverse spectrum of digital formats. GINA expert Ryan Hale explains, “The problem with all of these different systems is that they speak different languages. . . . GINA does not require the end systems to be modified to talk to them, GINA is built to understand all of the various data inputs or ‘languages’ and then creates links and relationships” [26].

As systems, networks, and data management grow increasingly complex, in turn, so do labor intensive and disparate systems. A colossal task for many organizations, especially Fortune 500 companies and the DOD, is setting up infrastructure to facilitate the sharing of copious amounts of data and resources with partners, clients, and providers. This desirable infrastructure should be ubiquitous, agnostic, support multiple users with differing semantics, accessible under multiple models for control, and agile [28].

GINA offers a multipurpose and innovative solution, Anderson describes how GINA provides a platform, facilities, and management of information and services across networks and all domains as “[GINA] can aggregate and objectify information from an unlimited set of heterogeneous information sources or service providers into a common information apparatus that can be tailored to specific system behaviors based on the relationship of the user to the information” [13].

Historically, interoperability and integration were seen as a task or problem of transferring data from one system into another. This was not a well-defined approach, and the first attempt at developing an approach was Enterprise Application Integration (EAI). EAI utilized an adaptor-mapping-based approach, in which adaptors understood how to communicate with one another using various standard systems. In effect, adaptors were integrated by wiring applicable components together. This four step approach could be described as: (1) adapt system A, (2) adapt system B, (3) map A to B, and (4) map B to A. This approach proved effective in managing two systems, however as environment became geometrically complex and the number of systems grew, it became evident that a composite approach was required to compensate for the addition of subsequent systems. A “Star integration” approach was created to support this expanding environment [27].

In a Star integration approach, a standard system is constructed in which all systems can communicate. This approach, even for two systems, is relatively more complex than the four-step approach described above. For example, the aforementioned four-step approach for integrating two systems, now becomes a seven-step approach: (1) adapt system A, (2) adapt system B, (3) define system C as the superset of A and B, (4) map A to C, (5) map C to A, (6) map B to C, and (7) map C to B. Extending this approach by adding a third system adds four more steps: (8) adapt D, (9) extend C to include D, (10) map C to D, and (11) map D to C. As the number of systems increases, this approach or model begin to pay dividends. For instance, for 20 systems, there are 85 steps versus 400 for the adaptor-mapping-based approach. Furthermore, the 85 steps do not introduce additional complexities of how to handle multiple systems updates [27].

Star integration functions as designed when the relationship between the datasets in A and the datasets in B are one-to-one. As the number of systems increases, the complexity of interaction increases. The Star integration does not account for interoperability problems such as accountability, summarization, analysis, navigation, and semantic interoperation. To counter this problem, a set of new tools was developed, such as data mining, Online Analytical Processing (OLAP); Extraction, Transformation, and Loading (ETL); Enterprise Information Integration (EII); virtual databases. Each tool provides a partial solution, so multiple tools are required to address a given organization's needs. This multi-tool solution presents a new problem of getting the interoperability tools to interoperate [27]. A universal solution to address an organization's interoperability requirements is dependent on the organization's needs. To formulate a near-universal solution, a good starting point is to redefine the problem. GINA's inherent interoperability provides logical and functional, advancements and advantages over traditional architectural models and approaches; see Figure 19 and Table 2.

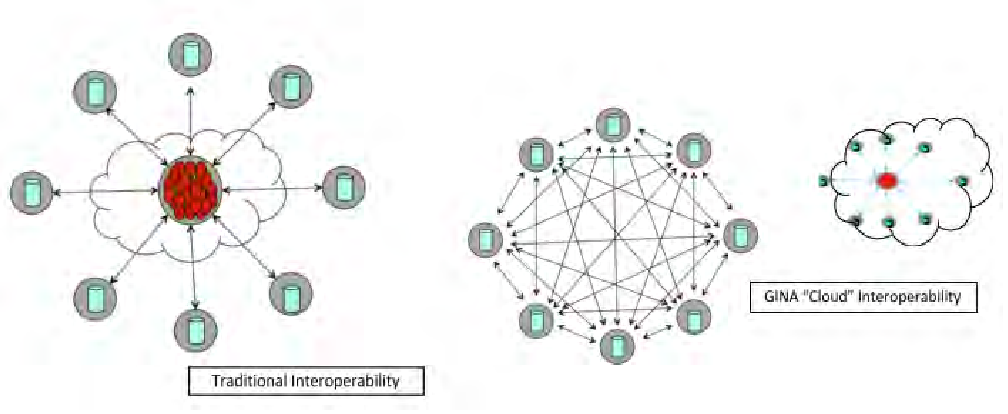
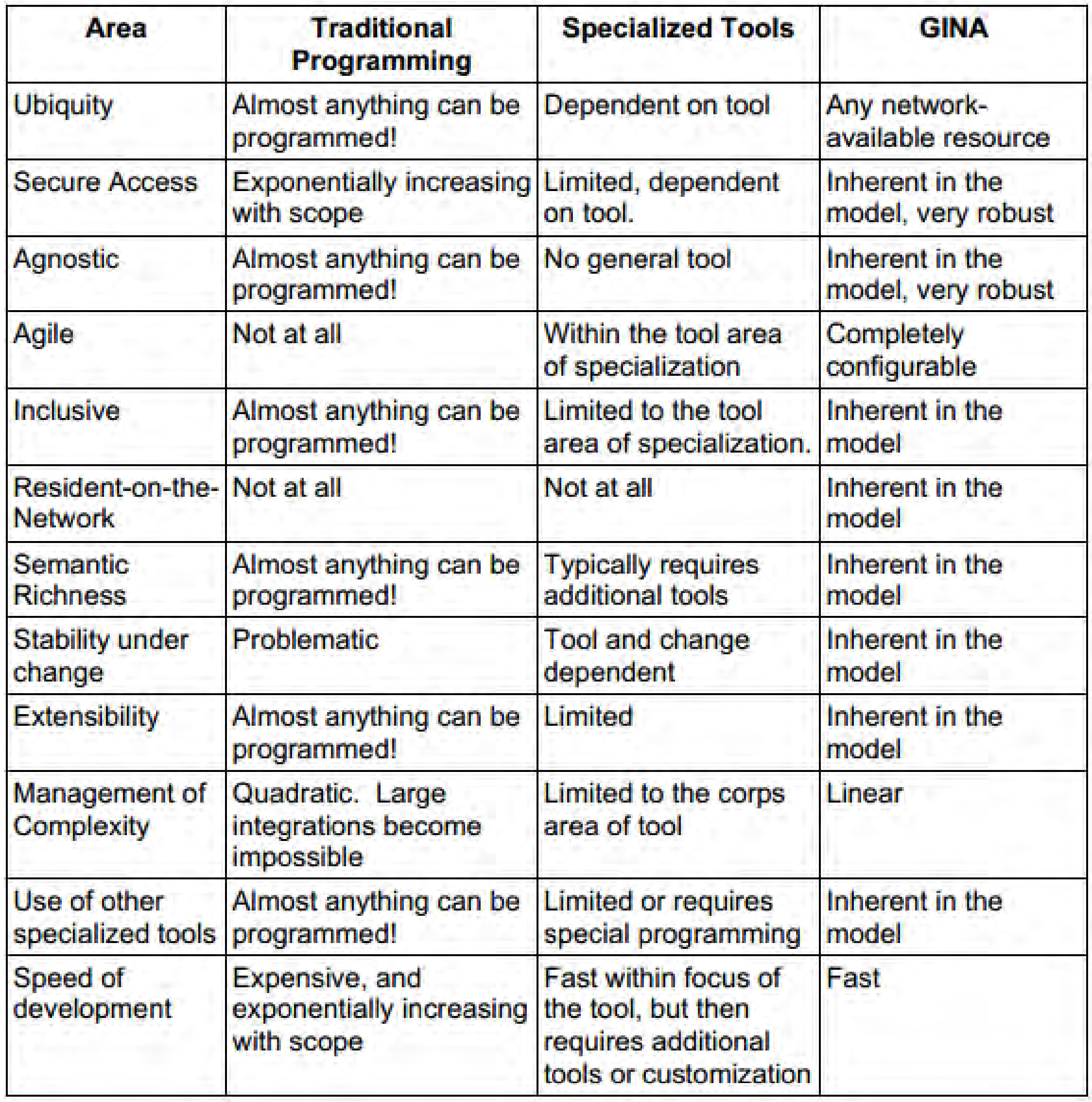


Figure 19. Service Oriented Architecture (SOA) versus GINA, from [27]. Each SoS is limited by a scalability problem of quadratic complexity. GINA interoperability does not exhibit this characteristic.



Area	Traditional Programming	Specialized Tools	GINA
Ubiquity	Almost anything can be programmed!	Dependent on tool	Any network-available resource
Secure Access	Exponentially increasing with scope	Limited, dependent on tool.	Inherent in the model, very robust
Agnostic	Almost anything can be programmed!	No general tool	Inherent in the model, very robust
Agile	Not at all	Within the tool area of specialization	Completely configurable
Inclusive	Almost anything can be programmed!	Limited to the tool area of specialization.	Inherent in the model
Resident-on-the-Network	Not at all	Not at all	Inherent in the model
Semantic Richness	Almost anything can be programmed!	Typically requires additional tools	Inherent in the model
Stability under change	Problematic	Tool and change dependent	Inherent in the model
Extensibility	Almost anything can be programmed!	Limited	Inherent in the model
Management of Complexity	Quadratic. Large integrations become impossible	Limited to the corps area of tool	Linear
Use of other specialized tools	Almost anything can be programmed!	Limited or requires special programming	Inherent in the model
Speed of development	Expensive, and exponentially increasing with scope	Fast within focus of the tool, but then requires additional tools or customization	Fast

Table 2. Comparison of SOA and GINA functionality, from [27].

APPENDIX B: AVERAGE TIME TO DETECT AND CONTAIN NETWORK INTRUSIONS

For many organizations, cyber attacks and intrusions go unnoticed for significant periods because the evidence of their activity is buried in thousands of lines of logs. According to Mandiant, an American cybersecurity firm that released a report accusing China of repeated cyber espionage intrusions, victim organizations failed to discover cyber attackers on their networks for an average of 229 days during 2013. This standard had improved from the median number of 243 days reported in 2012 [29]. The cybersecurity company Trustwave reported a median number of 87 days for an organization to detect an intrusion in 2014 and a median of 114 days from intrusion to containment [30]; see Figures 20 and 21.

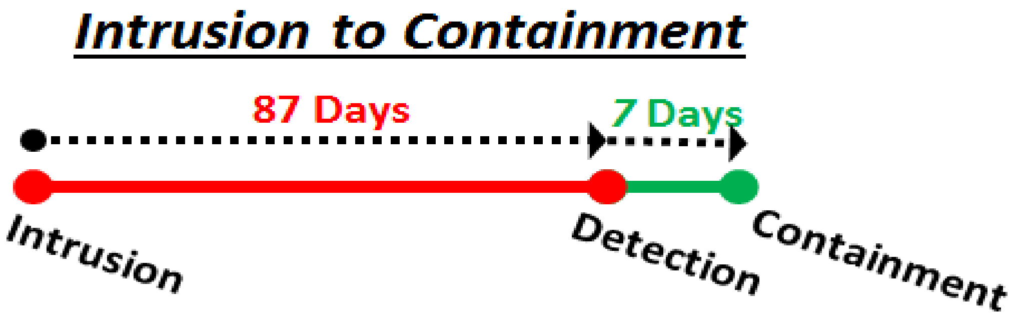


Figure 20. Median Number of Days between Intrusion to Containment, after [30]. Trustwave reports a median of 87 days before an organization detects a network breach. Generally, it requires another 7 days to contain the attack.

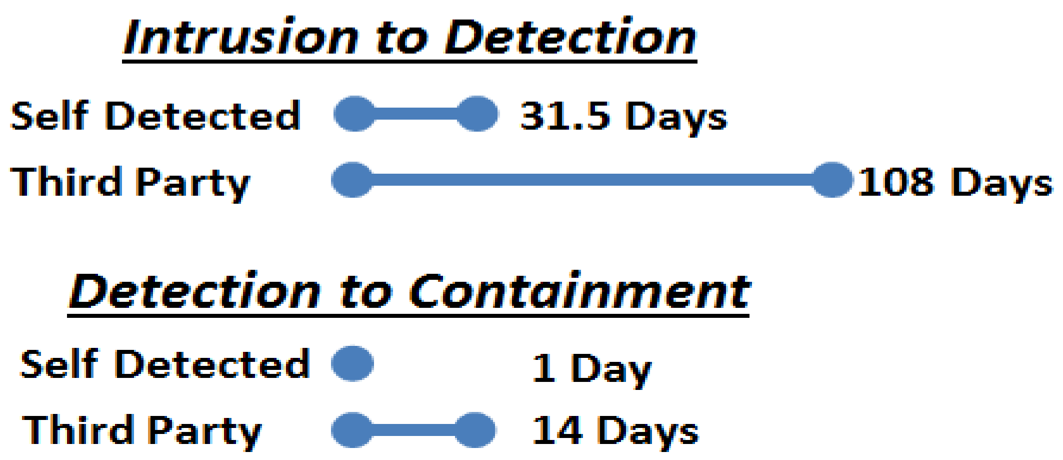


Figure 21. Median Number of Days between Intrusion to Detection to Containment for both Self Detection and Third Parties, after [30]. Organizations 'self-detecting' or internally detecting network intrusions significantly lowers the time required to detect and mitigate a cyber attack.

The majority of organizations struggle to find intrusions internally without the aid of external help from contracted cybersecurity specialists. In 2013, 33 percent of the organizations serviced by Mandiant detected the intrusion themselves, while 37 percent of the organizations had discovered their intrusions in 2012 [29]. Trustwave reveals a similar number, reporting that 71 percent of organizations fail to detect intrusions themselves [30]. In 2011, Verizon's annual Data Breach Investigations Report revealed that 92 percent of intrusions are not detected by the victim organization. This improved to 69 percent in 2013 [31]. Notably, external parties detect the majority of cyber breaches in an organization. Improving an organization's ability to internally detect cyber attacks significantly reduces the time to detect and mitigate them; see Figure 22.

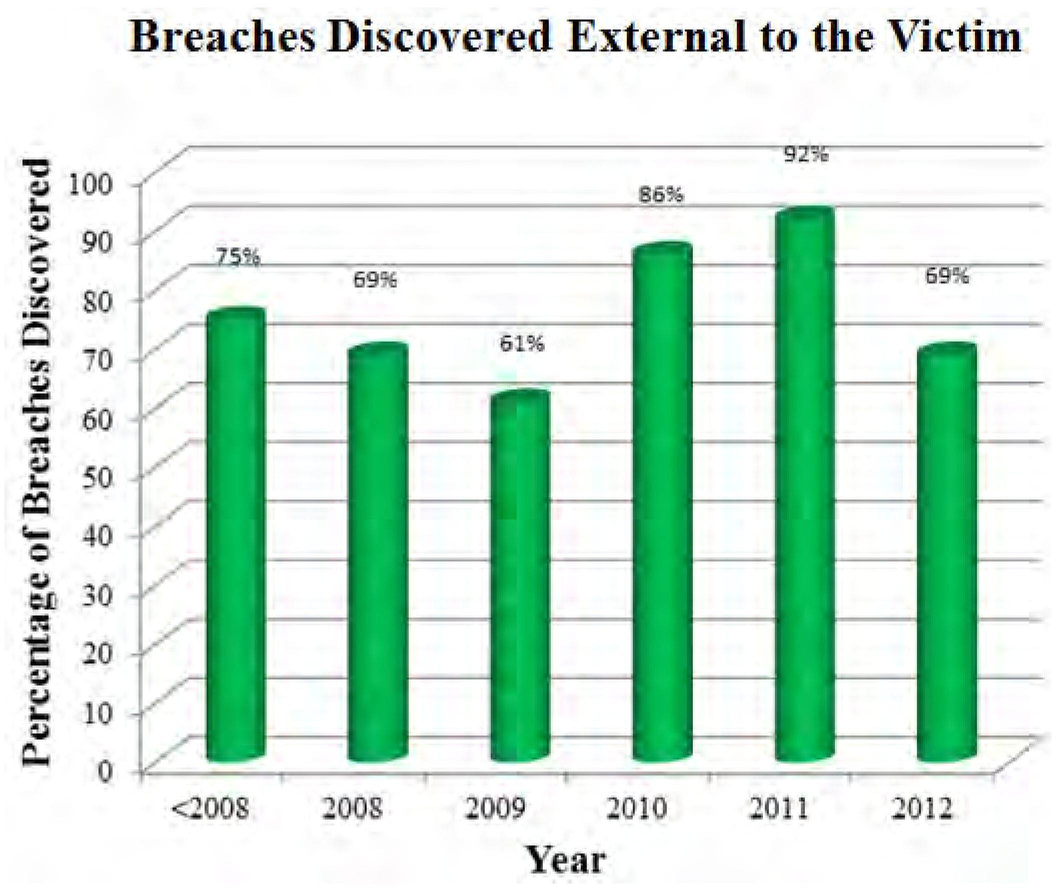


Figure 22. Percent of breaches discovered external of victim, after [31]. Improving an organization’s ability to internally detect cyber attacks significantly reduces the time to detect and mitigate them.

Reducing the time to detect an intrusion enables an organization to respond faster to an attack and limit its impact [9]; see Figures 23, 24, and 25. The time between the initial intrusion and detection allows an attacker to spread secretly, pivot, and execute an attack through an organization’s networks [9]. After the organization’s networks are logically and physically secure from the attacker, the resulting damage from the attack must be repaired.

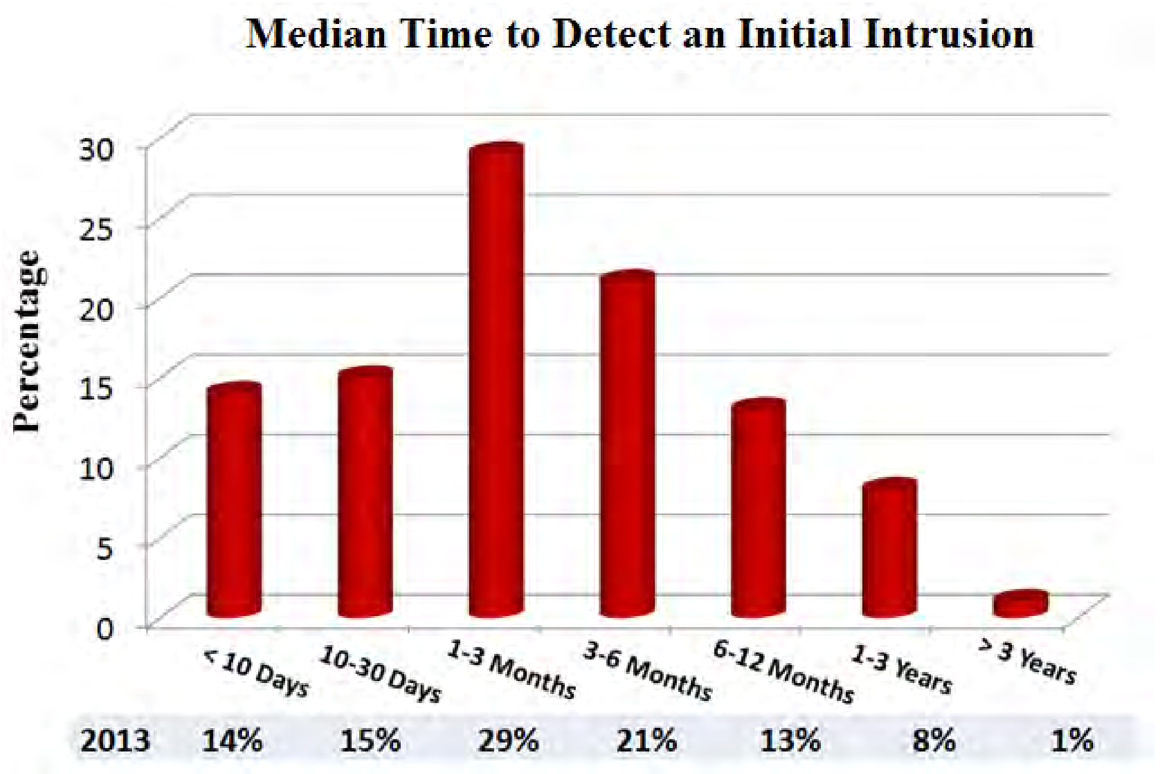


Figure 23. Median Time to Detect an Initial Intrusion, after [30]. Trustwave Holdings, an information security company serving 2.5 million business customers in 96 countries, compiled these figures by analyzing 691 data breach investigations conducted in 2013 from their global security operations centers located around the world.

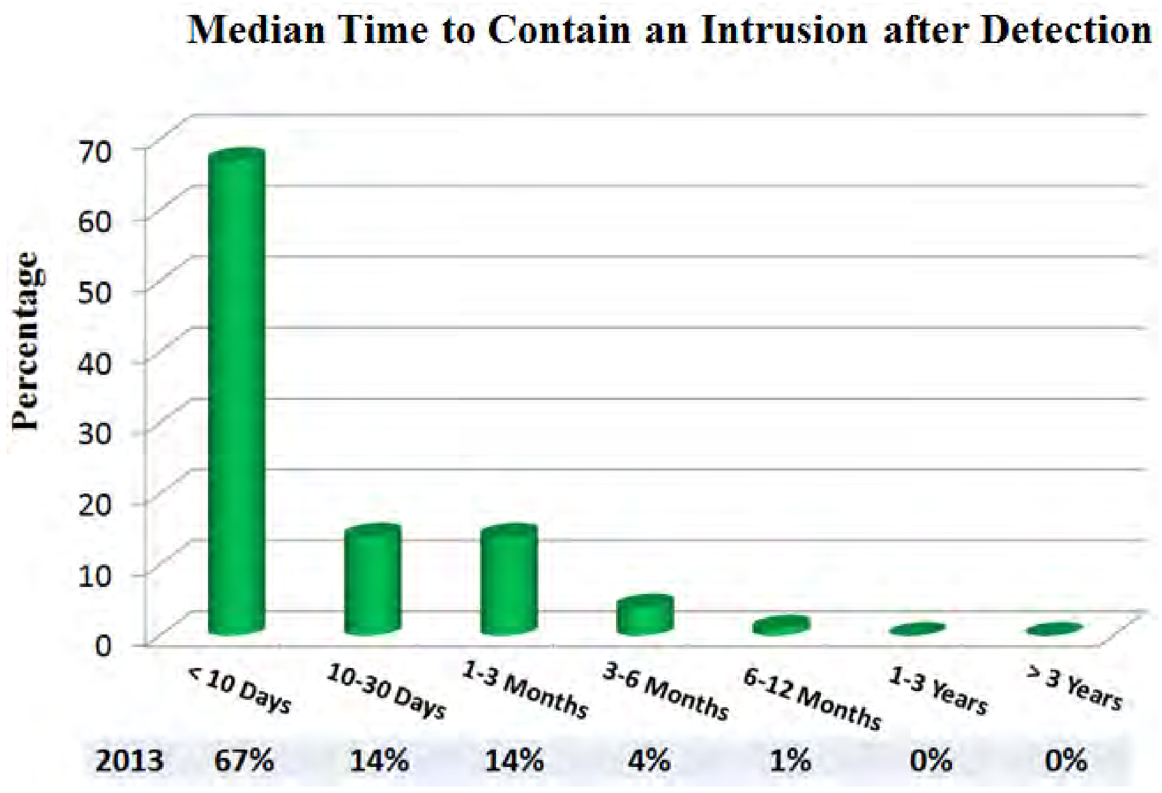
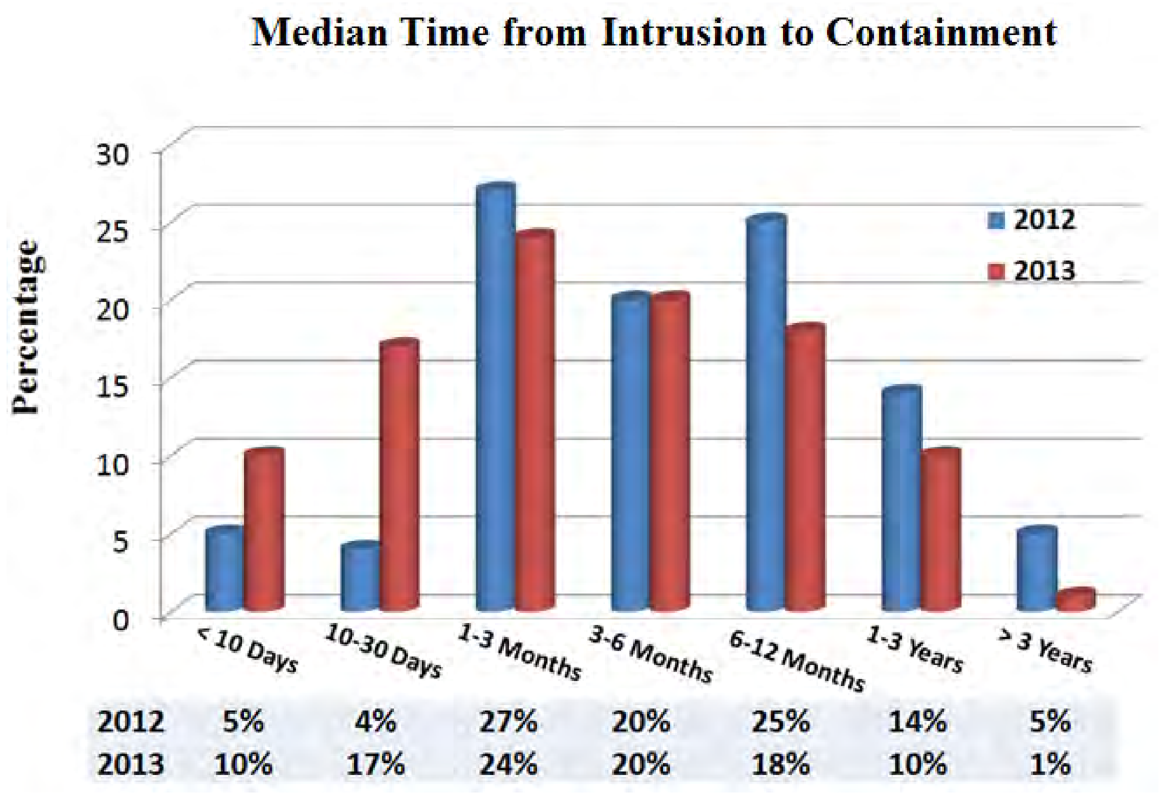


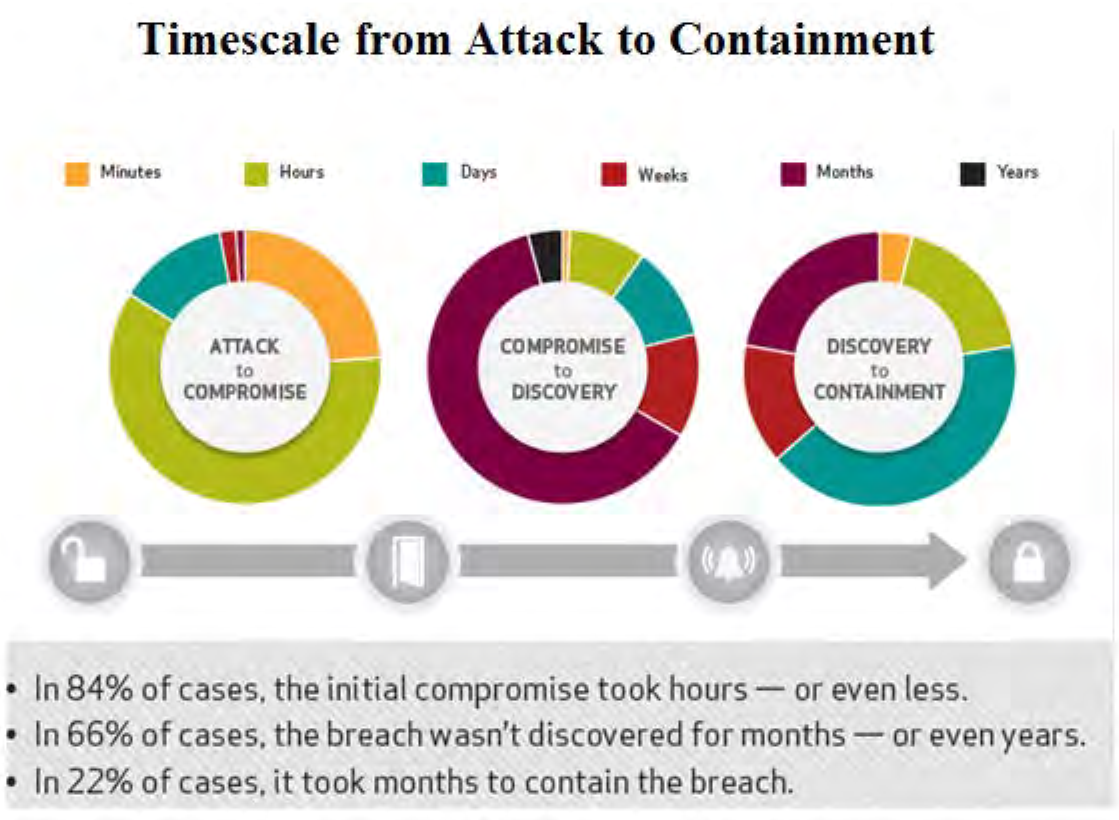
Figure 24. Median time to contain an intrusion after detection, after [30]. Trustwave Holdings, an information security company serving 2.5 million business customers in 96 countries, compiled these figures by analyzing 691 data breach investigations conducted in 2013 from their global security operations centers located around the world.



2012	5%	4%	27%	20%	25%	14%	5%
2013	10%	17%	24%	20%	18%	10%	1%

Figure 25. Median Time from Intrusion to Containment, after [30]. Trustwave Holdings, an information security company serving 2.5 million business customers in 96 countries, compiled these figures by analyzing 691 data breach investigations conducted in 2013 from their global security operations centers located around the world. For many organizations, cyber attacks and intrusions go unnoticed for significant periods of time.

Any delays in detecting, containing, mitigating, or purging the attacker from the organization’s networks gives the attacker more time to exploit the network or cause damage [32]. Organizations struggle to detect cyber attacks with many larger organizations taking months or years to detect intrusions [9]. In 2013, Verizon’s Data Breach Investigations Report revealed that 84 percent of network compromises took hours or less to commit [33]. The same report continues by stating that 22 percent of the time, containing these attacks takes months; see Figure 26. A delay in cyber defense creates opportunities for the attacker.



Timescale from Attack to Containment

- In 84% of cases, the initial compromise took hours — or even less.
- In 66% of cases, the breach wasn't discovered for months — or even years.
- In 22% of cases, it took months to contain the breach.

Figure 26. Timescale of Data Breaches, from [33].

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX C: DECISION-MAKING FRAMEWORKS

The DOD faces the major challenge of proactively managing information. In 2014, it is estimated that global data grows at 2.5 exabytes per day [34]. Organizations structured to create, update, organize, access, and transfer their collective body of knowledge to their operators and decision makers, provide a flexible decision-making process, moreover, an adaptable and extensible solution. Organizations are compelled to craft innovative solutions to combat challenges and threats to their security and regular, steady state operations in cyberspace. Organizations, specifically system engineers and administrators, could benefit extensively by implementing innovative technologies into the design, structure, and hardening of their networks. Taking the innovative possibilities of VRDM information modeling, the capstone team explored different decision-making frameworks and processes, to conceptualize and design a working model for our use case problem.

A. MULTI-CRITERIA DECISION MAKING

The team incorporated Multi-Criteria Decision Analysis (MCDA) into our SoS environment and VRDM construct. MCDA, also referred to as Multi-Criteria Decision Making (MCDM), is required to organize efficiently and effectively utilize data. MCDA is a related discipline and synthesis of mathematics, operations research, and operations science, in that it employs advanced analytical methods to arrive at optimal decisions and solutions. MCDA, when ideally applied to a given environment, will maximize profit, performance, and yield, and conversely minimize loss, risk, and cost of the desired objective.

MCDA evaluates various complementary, supplementary, and conflicting criteria in making a decision. Quantitative and qualitative measures such as costs and performance are inputs to the decision-making model. In the cyber domain, emerging technologies introduce both enhanced solutions as well as complex problems. For the latter, it is essential for users and stakeholders of a new, innovative technology to properly define and structure the problem, in order to accurately evaluate multiple criteria and make informed decisions. The National Research Council (NRC) codifies MCDA as "a deliberative process for bringing together disparate public and stakeholder views to help generate decision criteria, rank-order alternatives, deal with uncertainty in regard to competing objectives, and formulate management trade-offs between objectives in the context of risk" [35]. MCDA is a consolidation of methods employed to better comprehend, structure, and enhance decision processes. According to Linkov and Seager, conventionally, the MCDA process consists of the following four steps [35]:

- Structuring the problem by identifying and assessing various criteria through stakeholder elicitation that apply to the given decision.
- Eliciting model parameters, such as alternatives, decision criteria, relative weights, and preference threshold, and then evaluating the performance of each alternative on each criterion.
- Applying a decision algorithm that ranks each alternative from highest to lowest preference.
- Analyzing results of the model and repeating the process from Step 1 or Step 2 to re-evaluate the results.

This evaluation process organizes and synchronizes information from multiple sources, for example from various databases. It then streamlines complex, diverse, and potentially conflicting objectives to formulate an optimal solution for the decision maker. In Step 1 of the process, good decisions are framed by identifying and defining objectives; specifically, objectives that are "specific, measureable, realistic, and time-dependent" [36]. In Step 2, alternatives are identified, categorized, and classified to achieve the desired, defined objectives. Next, in Step 3, criteria are developed which are used to evaluate and assign values to each alternative according to the degree it contributes to reaching the set objectives. Also, in Step 3, a decision algorithm ranks the weighted alternatives against the cost-effectiveness of the competing criteria. Finally, Step 4 provides an alternative to the decision maker. However, even after the selection of an alternative, continual re-evaluation and further analysis can produce a superior alternative. MCDA is a reiterative process, and a built in feedback loop provides a forum to reassess past decisions continuously. In effect, this evaluation process systematically formalizes lessons learned and communicates them to decision makers enabling them to make better future decisions [36], [37]; see Figure 27.

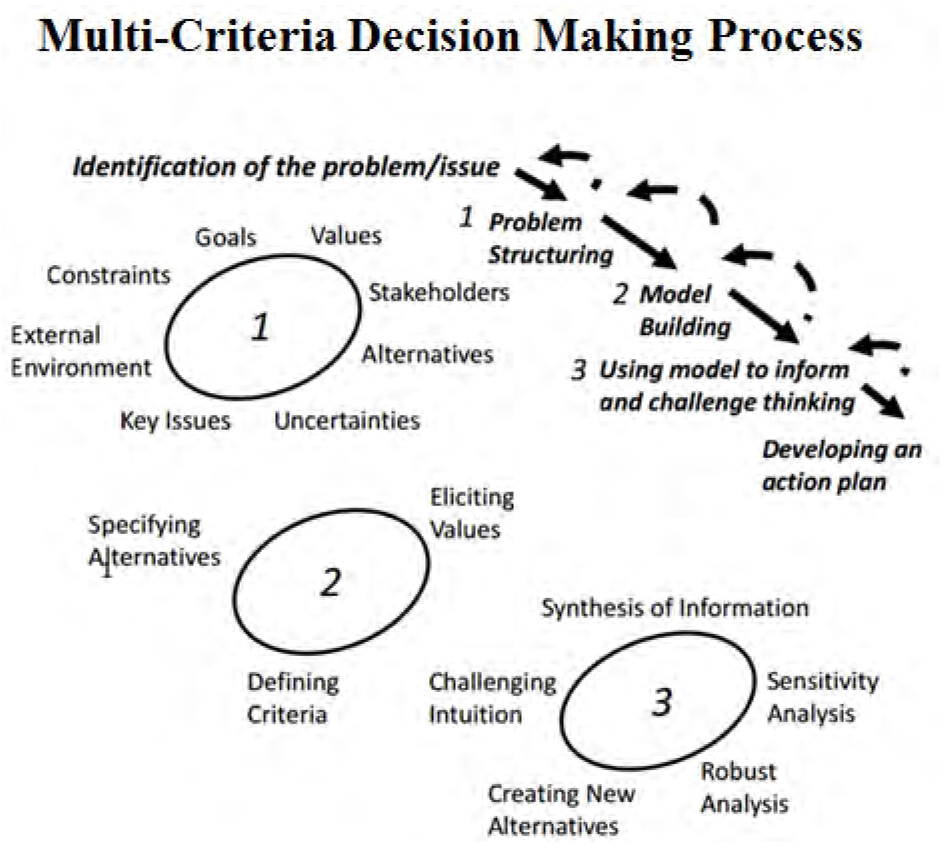


Figure 27. Multi Criteria Decision-making Process, after [36]. MCDA is a reiterative process, and a built in feedback loop provides a forum to reassess past decisions continuously. This evaluation process systematically formalizes lessons learned and communicates them to decision makers.

B. KNOWLEDGE MANAGEMENT

The Army Field Manual 6–01.1 describes KM as “the process of enabling knowledge flow to enhance shared understanding, learning, and decision-making” [38]. KM seeks the following objectives: 1) Shared knowledge by orientating and synchronizing people, processes, and tools within an organization. 2) Collaborative partnership that promotes flexibility, adaptability, and coordination of the shared knowledge to facilitate and expedite organizational decision-making. KM is not a technical system or a piece of software, but a person-driven process [39]. However, technical systems enable KM and significantly enhance an organization's ability to channel pertinent data to their decision makers; see Figure 28.

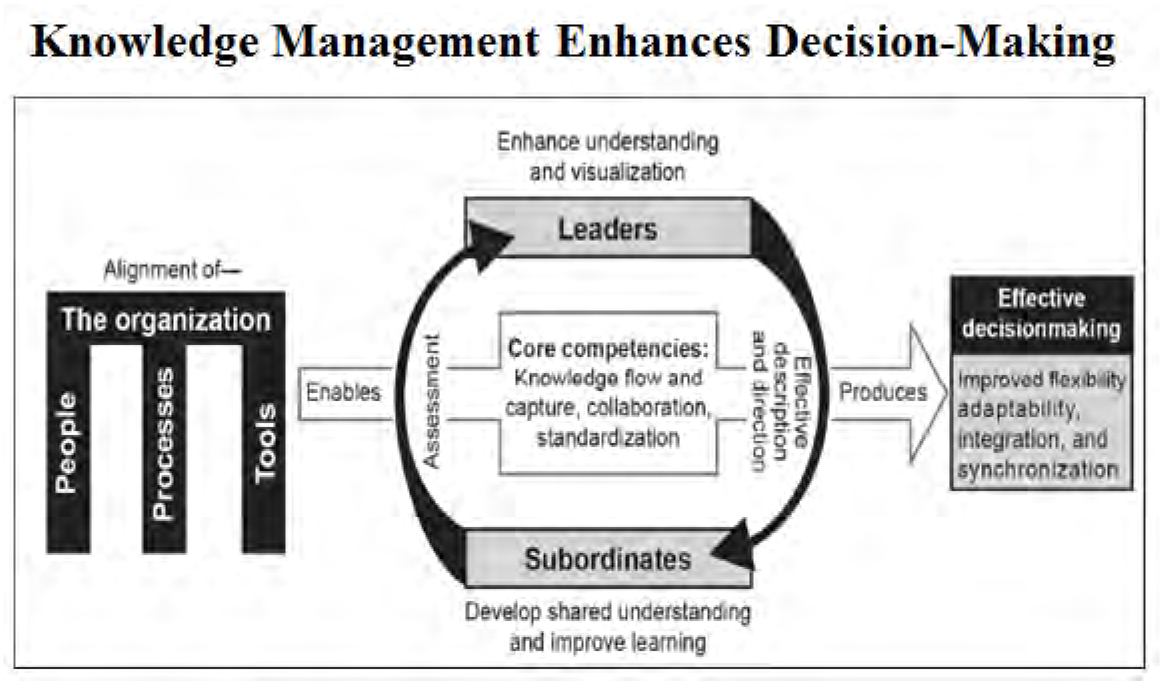

Knowledge Management Enhances Decision-Making
Enhance understanding and visualization
Leaders
Alignment of—
The organization
People
Processes
Tools
Enables
Assessment
Core competencies: Knowledge flow and capture, collaboration, standardization
Effective description and direction
Produces
Effective decisionmaking
Improved flexibility adaptability, integration, and synchronization
Subordinates
Develop shared understanding and improve learning

Figure 28. How Knowledge Management Enhances Decision-Making, from [38]. Technical systems enable KM and significantly enhance an organization's ability to channel pertinent data to decision-making.

There are five processes in KM used to achieve organizational goals [38]; see Figure 29.

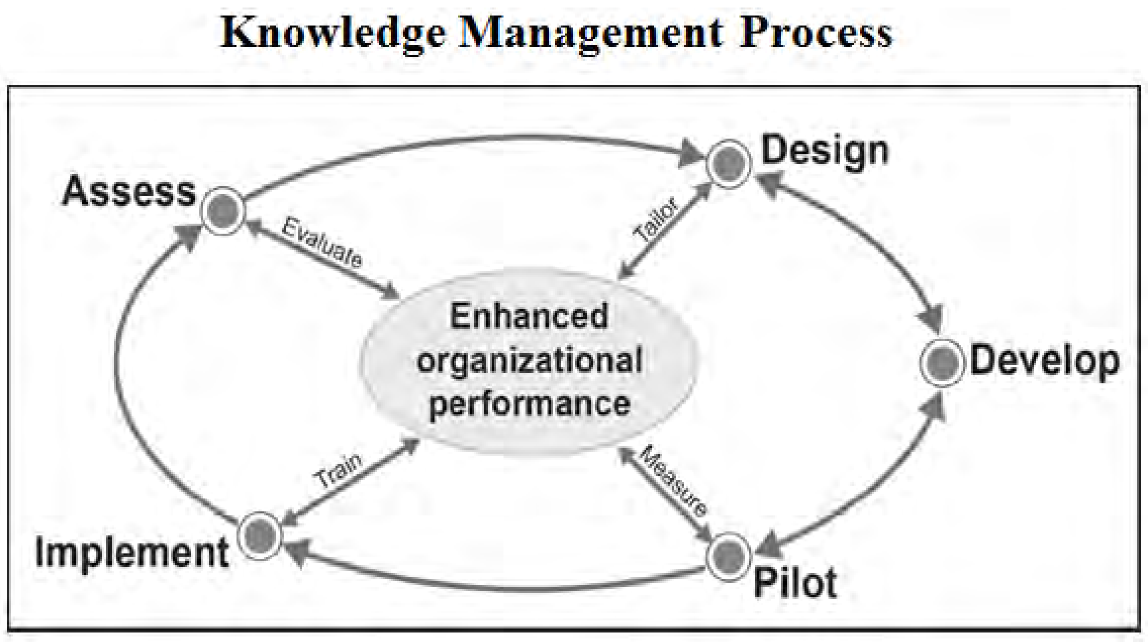

Knowledge Management Process
Assess
Design
Develop
Pilot
Implement
Enhanced organizational performance
Evaluate
Tailor
Measure
Train

Figure 29. Knowledge Management Process, from [38]. RAMPART, a technical KM solution, creates an information apparatus for KM by integrating multiple sources into a single framework allowing automated decision-making on the correlated data and subsequent system responses.

- Assess – This step evaluates the flow of data in the organization. It identifies obstacles to the free flow of data. It creates methods to mitigate or eliminate the obstructions.

- Design – A solution to improve the flow of data is developed. This step creates a strategy to improve the data transfer through products or processes. In a technical solution, the design phase would oversee the creation of the proposed information system.

- Develop – This step combines the products from the two previous steps of Assess and Design into an actionable solution appropriate for the organization. For a technical solution, this would be actual construction of the information system.

- Pilot – The solution from the Develop phase is created, deployed, and tested with organization. This solution is a test solution to identify any problems and to validate it as a KM solution for the organization.

- Implement – This final step sees the operational deployment of the solution including training operators and coaching personnel in their roles and responsibilities in using the system. Even after successfully deploying the solution, the system continues to be monitored and refined to assess and improve results.

At NPS, ITACS would benefit from a method that decreases the amount of time required for an operator to discover and respond to a cyber attack. The team implemented the following five steps of the KM process for the ITACS use case.

- Assess – The team observed and identified the needs and desires of the operators protecting their systems from malicious traffic. Their TTPs and Standard Operating Procedures (SOP) were consulted to help identify a technical solution to improving their data flow and enabling their operators' greater access to all the data and respond faster to a cyber attack.

- Design – Our RAMPART team brainstormed and outlined a system that would increase their access to the necessary data, to find an attack, and then react. Their current methods had operators crawling through numerous logs to compare data from different sensors and cybersecurity tools to first identify an attack. Comparing all the data was very time-consuming often causing many cyber attacks to go undiscovered for weeks. Contributing to the delay in identifying cyber attacks, ITACS was undermanned. ITACS needed a solution that brought all the data from the various logs into a easier to use format and ideally in one place, reducing the time spent searching the various, separate logs. The solution needed to decrease the necessary manpower needed to discover cyber attacks on the networks. Lastly, the solution needed to reduce the amount of time it took to respond to a cyber attack either through near real-time notification to the operator or through an automatic response to attacks in accordance with their established TTPs and SOPs.

- Develop – Our team created a system that delivers a KM solution by providing the operator aggregated data from his various cyber-defense systems. Data collected from these numerous sources would all feed a single aggregated module enabling the information apparatus to respond automatically to attacks identified per criteria preset by the operator.

- Pilot – Our proof of concept system, named RAMPART, is an implementation of a VRDM executable information apparatus in the GINA framework that is designed to collect data from the network and identify potential phpMyAdmin attacks using decision processes driven by real world operations procedures and SME decision-making.

- Operational Implementation – After validating the system through numerous tests and scenarios, RAMPART could be delivered as a viable solution to ITACS to improve their decision-making and cyber attack response.

RAMPART is a technical KM solution. It creates an information apparatus for KM through integrating multiple sources into a single framework allowing automated decision-making on the correlated data and subsequent system responses. It utilizes components of effective cyber-protection tools as components to streamline data flow to operators thus enhancing their decision-making. Incorporating data from multiple sensors, databases, cyber-tools, and processes, RAMPART creates an integrated model enabling operators and decision-making to identify and react faster to any cyber attack by automating log analysis.

C. RAMPART AS A DYNAMIC CYBER KM

The team integrated tenets of KM and MCDA into the elastic GINA framework. Integration was possible because the GINA framework creates an "object-oriented, software-based modeling environment for the modeling of various data sources and [allows] queries and transactions across those sources" [28]. Additionally, GINA optimizes "the use of existing systems, provides extensions to those systems, presents information to other systems and users from existing systems, maximizes the reuse of existing data and systems, and creates new uses and systems that seamlessly interoperate with existing environments" [28].

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX D: ITACS INCIDENT HANDLING PROCEDURES

NPS ITACS cyber incident handling employs the following five phases; see Figure 30.

- Identification; see Figure 31.
- Containment; see Figure 32.
- Neutralization; see Figure 33.
- Recovery; see Figure 34.
- Assessment and Documentation; see Figure 35.

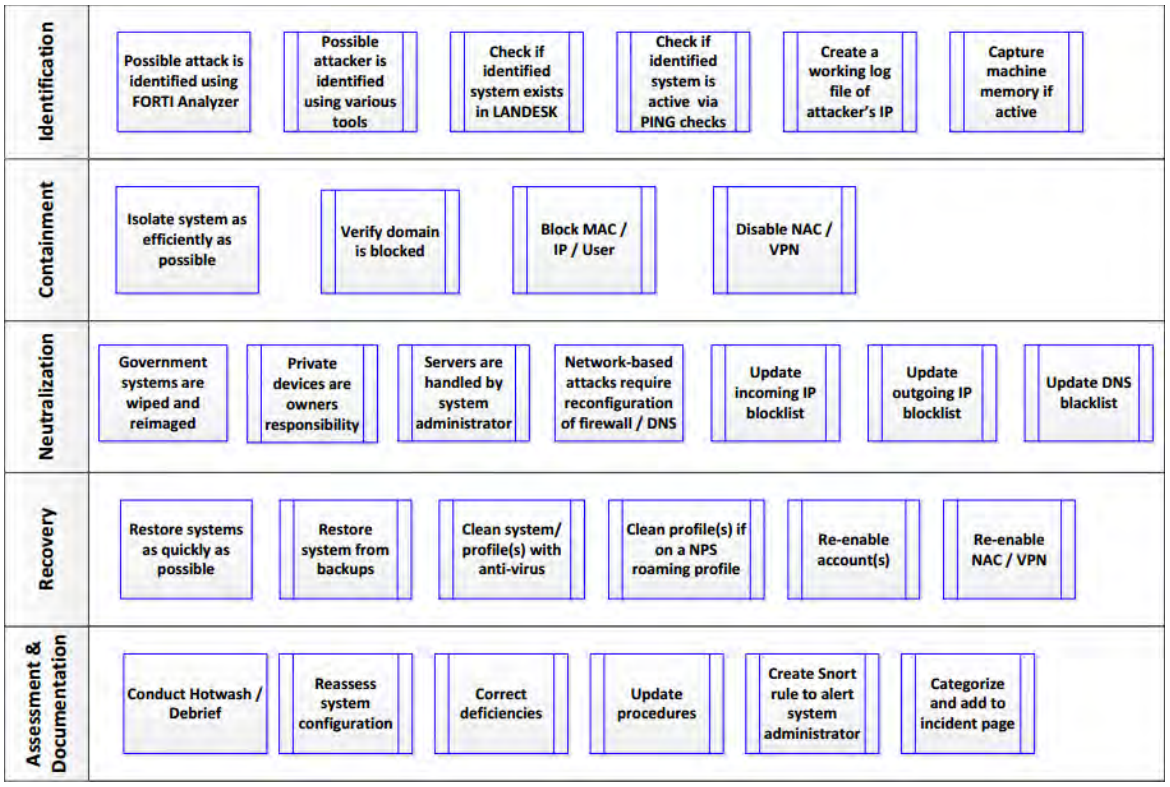

Identification
Possible attack is identified using FORTI Analyzer
Possible attacker is identified using various tools
Check if identified system exists in LANDESK
Check if identified system is active via PING checks
Create a working log file of attacker's IP
Capture machine memory if active
Containment
Isolate system as efficiently as possible
Verify domain is blocked
Block MAC / IP / User
Disable NAC / VPN
Neutralization
Government systems are wiped and reimaged
Private devices are owners responsibility
Servers are handled by system administrator
Network-based attacks require reconfiguration of firewall / DNS
Update incoming IP blocklist
Update outgoing IP blocklist
Update DNS blacklist
Recovery
Restore systems as quickly as possible
Restore system from backups
Clean system/ profile(s) with anti-virus
Clean profile(s) if on a NPS roaming profile
Re-enable account(s)
Re-enable NAC / VPN
Assessment & Documentation
Conduct Hotwash / Debrief
Reassess system configuration
Correct deficiencies
Update procedures
Create Snort rule to alert system administrator
Categorize and add to incident page

Figure 30. ITACS Incident Handling Process – Overview.

A. IDENTIFICATION

Operators follow these basic procedures:

- Use FortiAnalyzer, a tool that records the source and destination IP addresses and identifies attacks via pattern and characteristic recognition. Look up webfilter logs for the IP identified as the non-DOD (Hostile) IP for a time period surrounding the time of the event.

- Attach those logs in the FortiAnalyzer tab of the IACD issue created for this incident.

- Look up the non-DOD IP in IP Audit (for the time period of the attack)

- Attach those logs under IP Audit.

- If Snort generated a real-time report on this incident, include the Snort Alert in the "Other Logs" tab.

- Use a reverse Domain Name System (DNS) to look up the URL and other information regarding the non-DOD IP.

- Capture the information in the non-DOD IP block of the issue. If the information is hosted on a webpage, a link will suffice.

- Document the IP(s) at NPS that were affected by the attack.

- Use the attack classification provided by the alerting sensor to name the attack carried out, and label the issue appropriately.

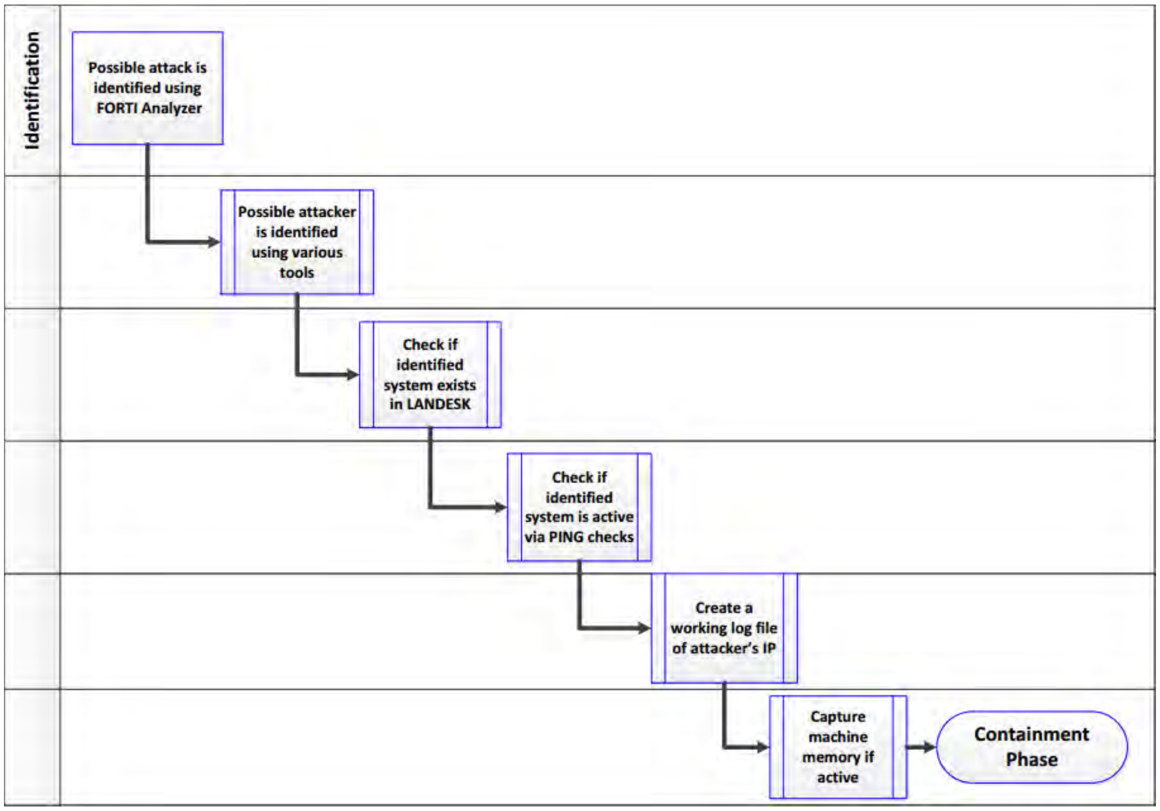

Identification
Possible attack is identified using FORTI Analyzer
Possible attacker is identified using various tools
Check if identified system exists in LANDESK
Check if identified system is active via PING checks
Create a working log file of attacker's IP
Capture machine memory if active
Containment Phase

Figure 31. ITACS Incident Handling Process – Identification Phase.

B. CONTAINMENT

Operators follow these basic procedures:

- SafeConnect (Preferred for end users)
- If possible, use the machine's MAC address to block, as it is unique to an affected machine and will not lead to incorrect blocks.
- If the MAC cannot be identified, use the IP Address.
- Failing 1 and 2, block the user by username.
- Servers shall be blocked.
- In addition, end users on the infected system(s) are directed to change their passwords.

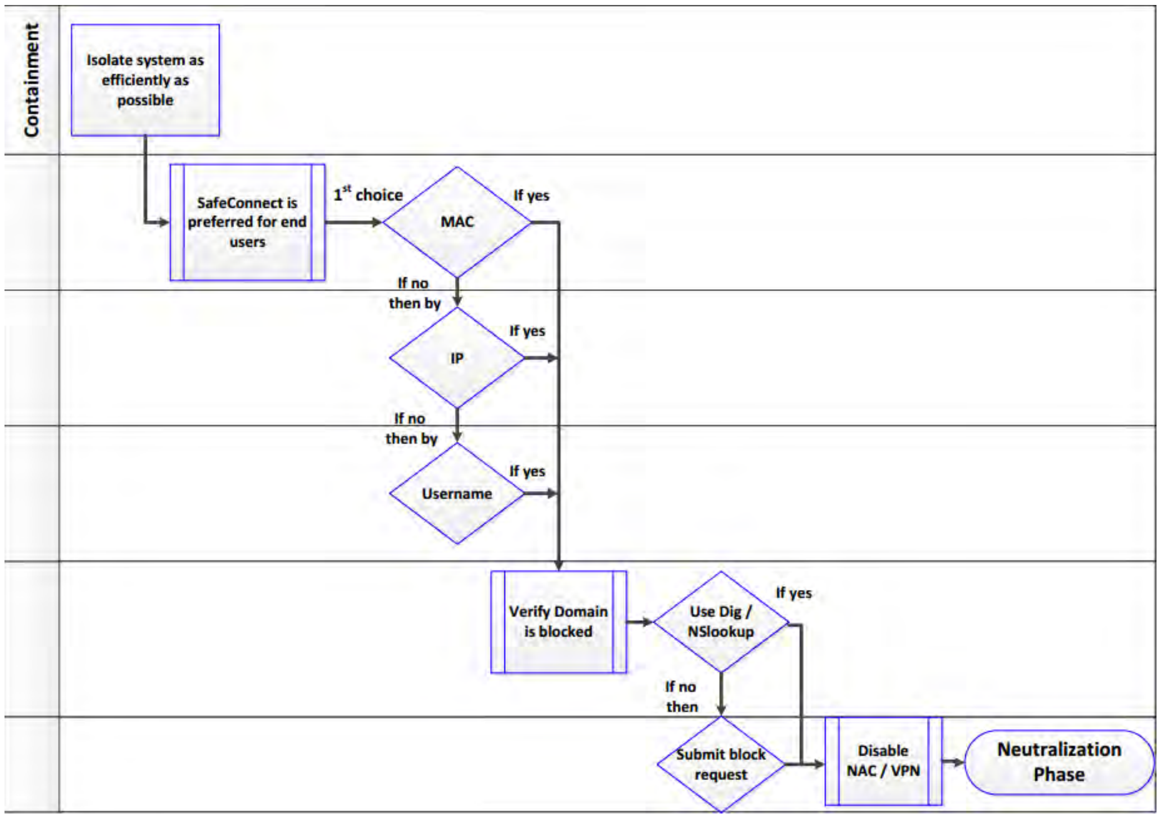


Figure 32. ITACS Incident Handling Process – Containment Phase.

C. NEUTRALIZATION

Operators follow these basic procedures:

- Is the system a Government Computer (if so re-image).
- If it is a personal system send instruction on how to clean.
- Is the system a server (servers are handled by System Administrators only).
- Was it a network based attack?
- If yes to step 4 then Access Control List (ACL) and DNS lists must be updated.

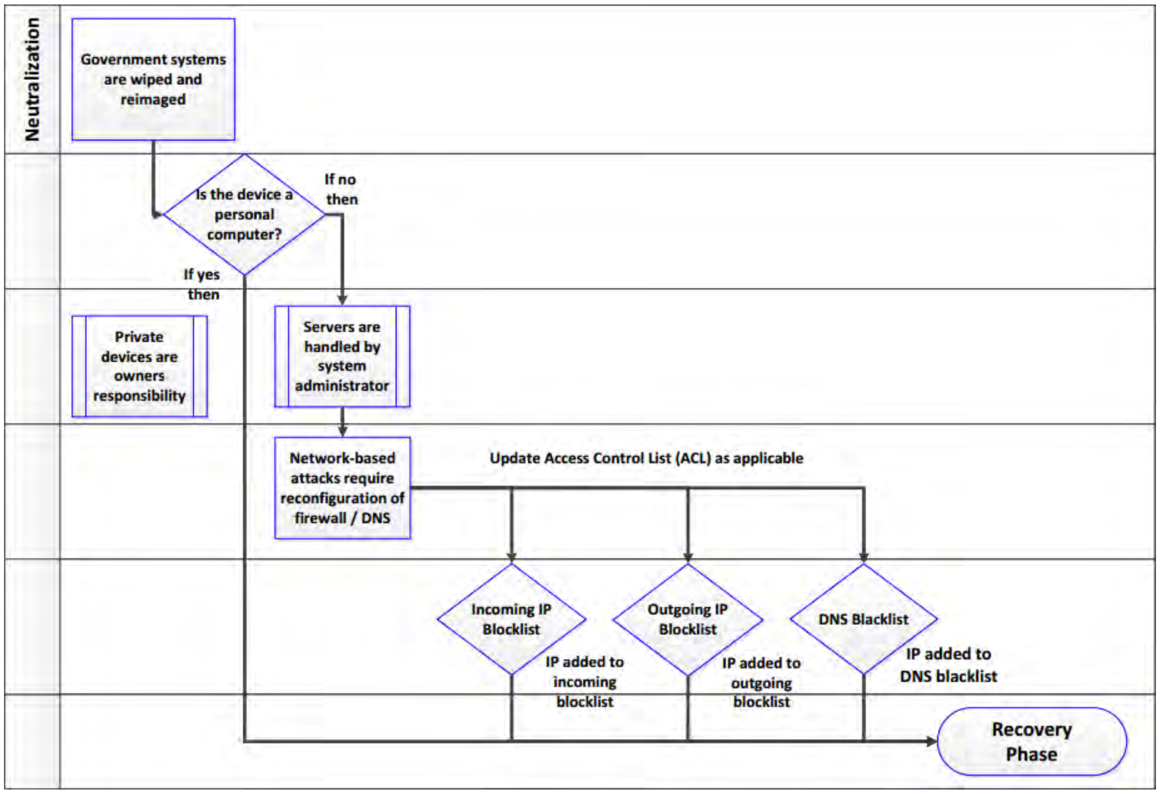

Neutralization
Government systems are wiped and reimaged
Is the device a personal computer?
If no then
If yes then
Private devices are owners responsibility
Servers are handled by system administrator
Network-based attacks require reconfiguration of firewall / DNS
Update Access Control List (ACL) as applicable
Incoming IP Blocklist
Outgoing IP Blocklist
DNS Blacklist
IP added to incoming blocklist
IP added to outgoing blocklist
IP added to DNS blacklist
Recovery Phase

Figure 33. ITACS Incident Handling Process – Neutralization Phase.

D. RECOVERY

Operators follow these basic procedures:

- Identify how far back you need to go (time of infection, what system is affected).
- Compare with the available backup media (dates and systems available).
- Perform restoration that meets the required coverage.
- Complete restoration by updating the system to bridge the gap from restore the point to operational status.
- Re-image hard drive from known clean image. Do not attempt to "clean" the malware from the system.

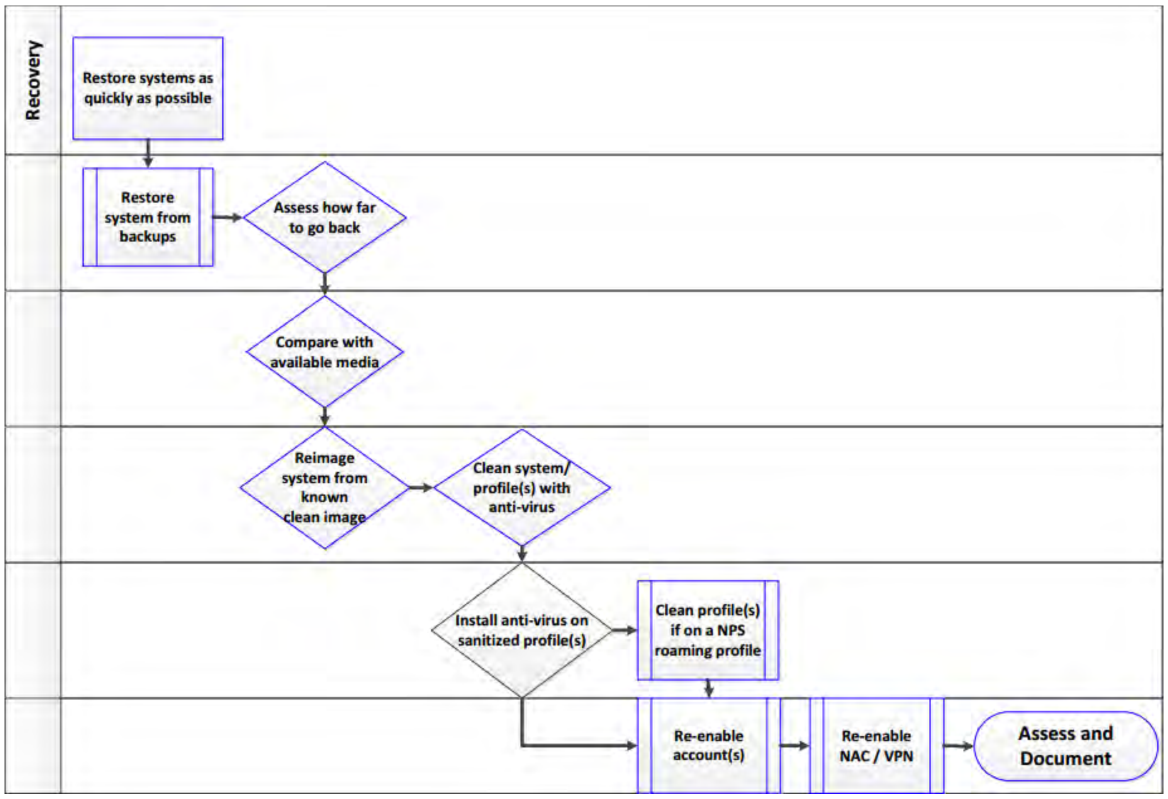

Recovery
Restore systems as quickly as possible
Restore system from backups
Assess how far to go back
Compare with available media
Reimage system from known clean image
Clean system/ profile(s) with anti-virus
Install anti-virus on sanitized profile(s)
Clean profile(s) if on a NPS roaming profile
Re-enable account(s)
Re-enable NAC / VPN
Assess and Document

Figure 34. ITACS Incident Handling Process – Recovery Phase.

E. ASSESSMENT AND DOCUMENTATION

Operators follow these basic procedures:

- Conduct a debrief on the incident.
- Reassess the system configuration for possible changes.
- Correct deficiencies and update procedures.
- Create a Snort rules to alert Admins of future incidents
- Categorize and add the incident to the incident page.

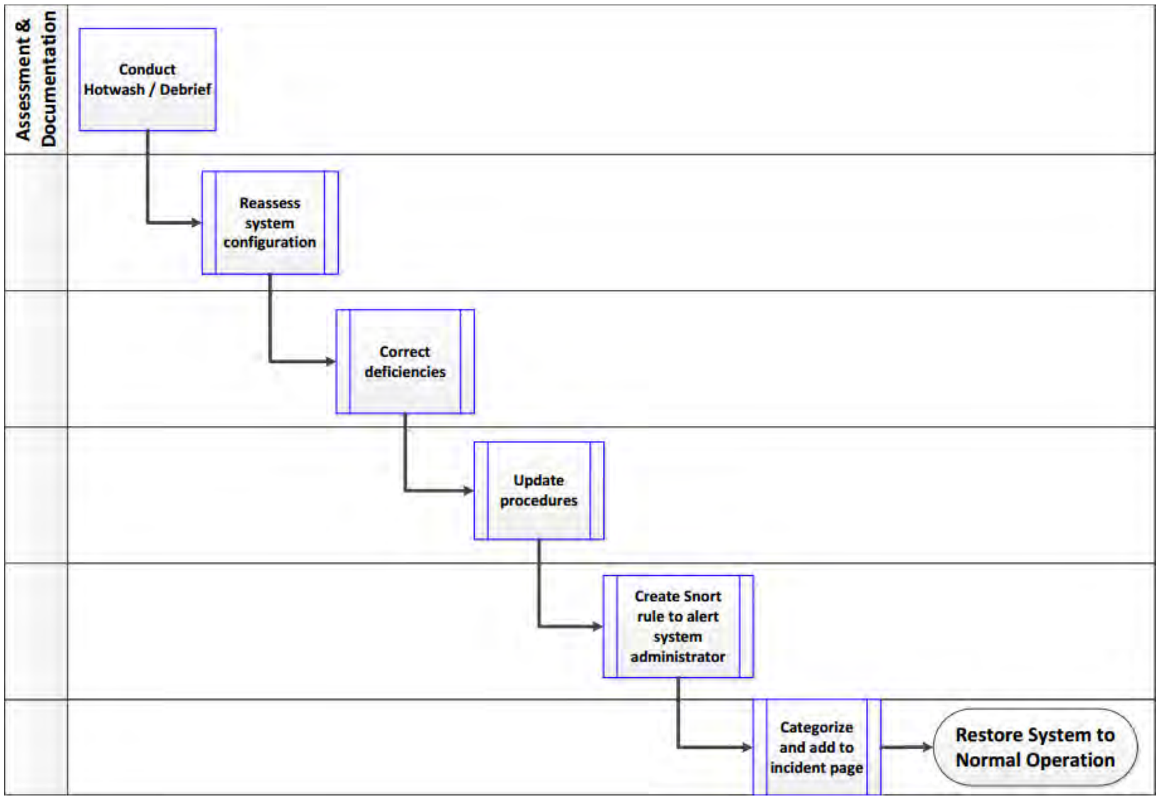

Assessment & Documentation
Conduct Hotwash / Debrief
Reassess system configuration
Correct deficiencies
Update procedures
Create Snort rule to alert system administrator
Categorize and add to incident page
Restore System to Normal Operation

Figure 35. ITACS Incident Handling Process – Assessment and Documentation Phase.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX E: RAMPART TECHNICAL IMPLEMENTATION

RAMPART implements an automated cyber threat response model prototype that demonstrates an intrusion protection capability within GINA. GINA has no published history of implementation as an IPS.

A. RAMPART DECISION MODEL

In order to establish a threat level, the model must maintain state with each network event. An event is generally a service request from an IP address to a certain server socket at a certain time. RAMPART uses the remote IP address as the identity. The state table exists in the information apparatus database on the GINA server. The result of each test is recorded as a binary value maintained within a four-tuple in context of IP address. The four-tuple is {IP, Test1Result, Test2Result, Test3Result}.

The analysis and decision consists of three tests against an Apache web log event:

- Request for "phpMyAdmin." phpMyAdmin is a database administration service common to Apache web servers. Access to this site is generally reserved for database administrators. We assume that the database administrator will not access the site while simultaneously using Tor or a website scanning tool.
- "404" page not found status errors in a web server log. When these errors exist in rapid secession (more than one with identical source IP and timestamp to the second), it is evidence of automated activity. We assume that scanning activity is not always bad. For instance, search engine spiders and vulnerability assessment scanners may not be a reason for alarm, but could both generate many rapid "404" page not found status errors.
- Traffic with a source IP address of a Tor exit node. Tor traffic is not always malicious, e.g., there are instances where privacy is a legitimate concern. In such cases, Tor is a legal means to achieve that end. However, we assume that a malicious actor will obfuscate his IP address during a cyber attack to avoid attribution.

Triggering on any one of the events alone would yield unacceptable numbers of false positives, and yield an untenable workload for the analyst. Combining the "votes" though yields a decision scheme that correlates the indicators, allows weighting and simulates an analyst's decision process based on sufficient indicating criteria.

There are three possible outcomes of the decision model.

- Someone is scanning the network and has invoked phpMyAdmin at the same time.
- Someone is accessing phpMyAdmin, specifically from the Tor network.
- Someone is scanning from the Tor network.

B. ARCHITECTURE

We used a database logging solution for an Apache web server hosting a phpMyAdmin website without any other form of intrusion protection. It is important to note that GINA natively operates on a Microsoft Windows server that uses IIS to host a DotNetNuke environment connected to a MS SQL backend database. The implication is that a configured GINA information apparatus will aggregate a MS SQL database from a Windows server and a MySQL database from a Linux server to implement an all-source common cyber picture. A further implication is that the model is inherently extensible and subject to reconfiguration based on use case evolution, accommodating changes in mission, technology, and threat changes.

Apache Database Logging

We chose Apache as a network attack target to demonstrate that suspicious requests to phpMyAdmin can be investigated computationally by modeling a cyber threat analyst's decision-making process. The majority of web servers are Linux-based and we infer that there is a greater attack surface for Linux-based web servers. The threat detection model may also be applied to Microsoft IIS servers, but it would limit platform diversity for the demonstration. RAMPART monitors the Apache server loaded with phpMyAdmin as a honeypot.

Inputs

We use a GINA model to access a MySQL database that stores all logs from an Apache web server. The Apache log table contains four columns of interest: Remote IP, URL, Status, and Timestamp. We create GINA X-Type elements, the key model concepts, for each of the source columns necessary to perform an automated analysis of the data within. The SQL server search processing overhead of queries remains on the source MySQL server; the GINA model only reads in records that are specifically requested for the analysis. The GINA model will use string compares against the columns to determine the truth table that was outlined in the capstone scope.

Signal

The time interval for analysis of logged data is one minute. We determined that a batch analysis at regular time intervals is more efficient than performing an analysis of each log entry as it is created. Attacks can occur at such rapid succession that the server can be overwhelmed with duplicate processes. Batch analysis in one process eliminates the attacker’s capability to perform a denial of service on the GINA server by overwhelming it with false positives.

Threat Monitor

A separate model process checks the values in the state table to determine if sufficient threat level exists to invoke an automated response. The threat monitor is part of a GINA “service chain” which begins after each minute interval analysis process completes. The threat monitor invokes a response with the source IP address as an input and also marks each record for deletion after it has been processed. The automated response invokes a special form that initiates a command line with arguments. This allows any command line process to be initiated from the model.

Responder

A GINA model may also be configured to execute Java applets, run shell commands, and send email. In addition to command line execution, a GINA model can append as many command line arguments as desired. In this case, the automated response consists of a command line execution of a batch file with the attacker’s IP address as the argument for each automated response.

We use SSH scripting to communicate with external network hardware. Routers, servers, firewalls, and IPS generally support SSH as a secure management protocol. SSH commands can perform multiple security functions including:

- Modifying firewall rules to block an offending address.
- Updating a routing table to redirect an attacker to a honeypot.
- Disabling certain protocols on an interface.

SSH scripting can be implemented in a windows batch file by using a command line SSH program called Plink. Plink uses the IP address provided by the responder as an input and executes a command on the network router to block the attacker’s IP address. The threat monitor initiates a response via invocation of a GINA form (Responder) that receives the attacker’s IP address and initiates a command line execution of a batch file with the attacker’s IP address as a command line argument. The form also submits an email to the network administrator to inform him of the blocked IP address.

APPENDIX F: RAMPART INSTALL PROCEDURE

The cyber threat detection and response test range use both Linux and Microsoft servers. They are loaded in the following configurations: IIS (MS 2008), Apache RHEL 6, MS SQL (MS 2008), MySQL (RHEL 6), and Vyatta (Debian Linux). Server installations assume base OS and SQL server installations have been completed by the user. The GINA and Apache portions of this appendix are a modified excerpt from the 2014 thesis "Automated Cyber Threat Analysis and Specified Process Using Vector Relational Data Modeling" [1].

A. APACHE DATABASE

Apache database enabling programs are mod-dbd, apr-dbd, mod-log-dbd, and mod-vhost-dbd. Together the modules send server transaction logs to a MySQL database. Install procedure from RHEL 6 terminal:

1. Subscribe to Software Repositories

- yum install http://centos.alt.ru/repository/centos/5/x86_64/apr-util-1.4.1-1.el5.x86_64.rpm
- yum install http://yum.jasonlitka.com/EL5/x86_64/apr-util-1.3.9-1.jason.2.x86_64.rpm
- yum install http://mirror.centos.org/centos/6/os/x86_64/Packages/apr-util-mysql-1.3.9-3.el6_0.1.x86_64.rpm

2. Install Software Dependencies with the Following Commands in a Terminal Emulator

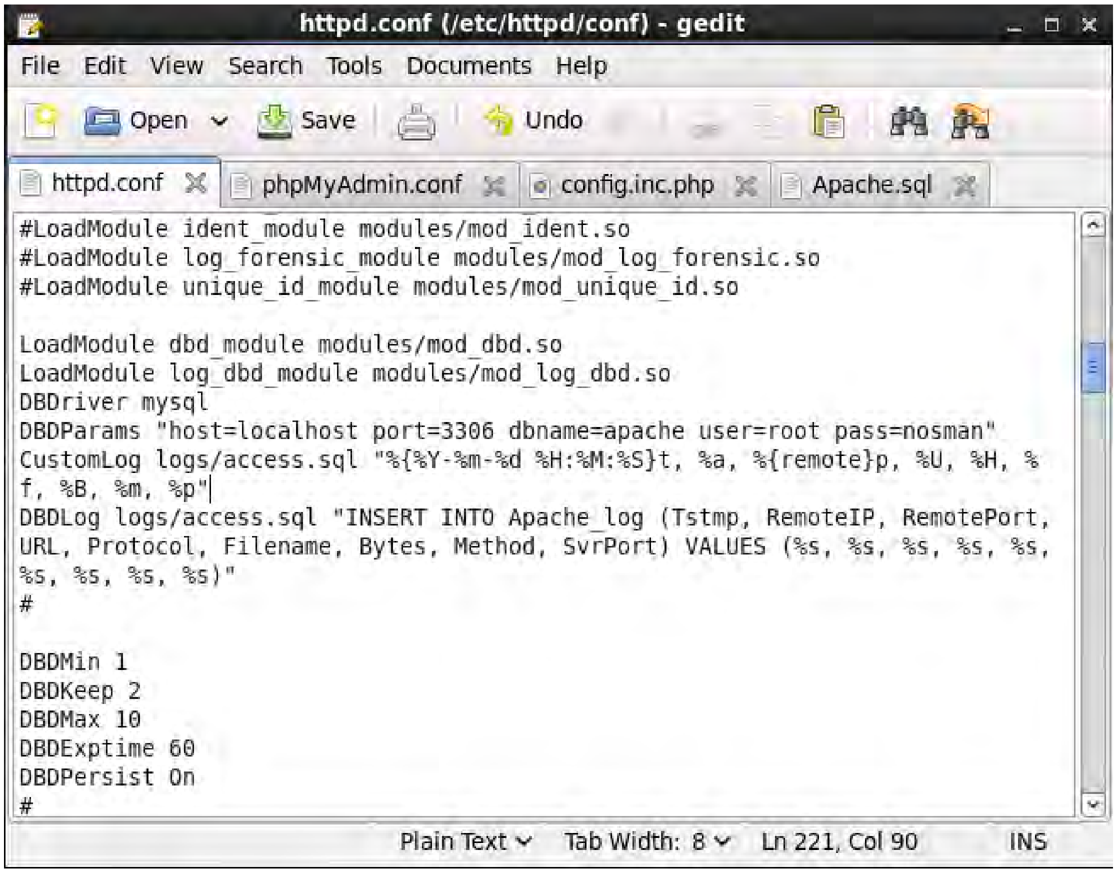
- yum install httpd-devel –y
- yum install mysql-server mysql –y
- yum install phpMyAdmin –y
- yum upgrade mysql –y
- yum install httpd –y

- yum install python-abi –y
- yum install python-crypto –y
- yum install python-paramiko –y
- yum install ftp://ftp.muug.mb.ca/mirror/centos/6.5/os/x86_64/Packages/ libzip-0.9-3.1.el6.x86_64.rpm -y
- yum install http://dev.mysql.com/get/Downloads/MySQLGUITools/ mysql-workbench-community-6.0.9-1.el6.x86_64.rpm -y
- wget https://dbd-modules.googlecode.com/files/dbd-modules-1.0.6.zip unzip dbd-modules-1.0.6.zip
- apxs -c mod_vhost_dbd.c
- apxs -i mod_vhost_dbd.la
- apxs -c mod_log_dbd.c
- apxs -i mod_log_dbd.la

3. Configure httpd.conf – The Sequential Order of the Directives Matter

- LoadModule dbd_module modules/mod_dbd.so
- LoadModule log_dbd_module modules/mod_log_dbd.so
- DBDriver mysql
- DBDParams "host=localhost port=3306 dbname-Apache user=root pass=nossman"
- CustomLog logs/access.sql "%{%Y-%m-%d %H:%M:%S}t, %a, %{remote}p, %U, %H, %f, %B, %m, %p, %>s"
- DBDLog logs/access.sql "INSERT INTO Apache_log(Tsp, RemoteIP, RemotePort, URL, Protocol, Filename, Bytes, Method, SvrPort, Status) Values (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s)"

  An example of the httpd file is below; see Figure 36.



```
#LoadModule ident_module modules/mod_ident.so
#LoadModule log_forensic_module modules/mod_log_forensic.so
#LoadModule unique_id_module modules/mod_unique_id.so

LoadModule dbd_module modules/mod_dbd.so
LoadModule log_dbd_module modules/mod_log_dbd.so
DBDriver mysql
DBDParams "host=localhost port=3306 dbname=apache user=root pass=nosman"
CustomLog logs/access.sql "%{%Y-%m-%d %H:%M:%S}t, %a, %{remote}p, %U, %H, %
f, %B, %m, %p"
DBDLog logs/access.sql "INSERT INTO Apache_log (Tstmp, RemoteIP, RemotePort,
URL, Protocol, Filename, Bytes, Method, SvrPort) VALUES (%s, %s, %s, %s, %s,
%s, %s, %s, %s)"
#

DBDMin 1
DBDKeep 2
DBDMax 10
DBDExptime 60
DBDPersist On
#
```

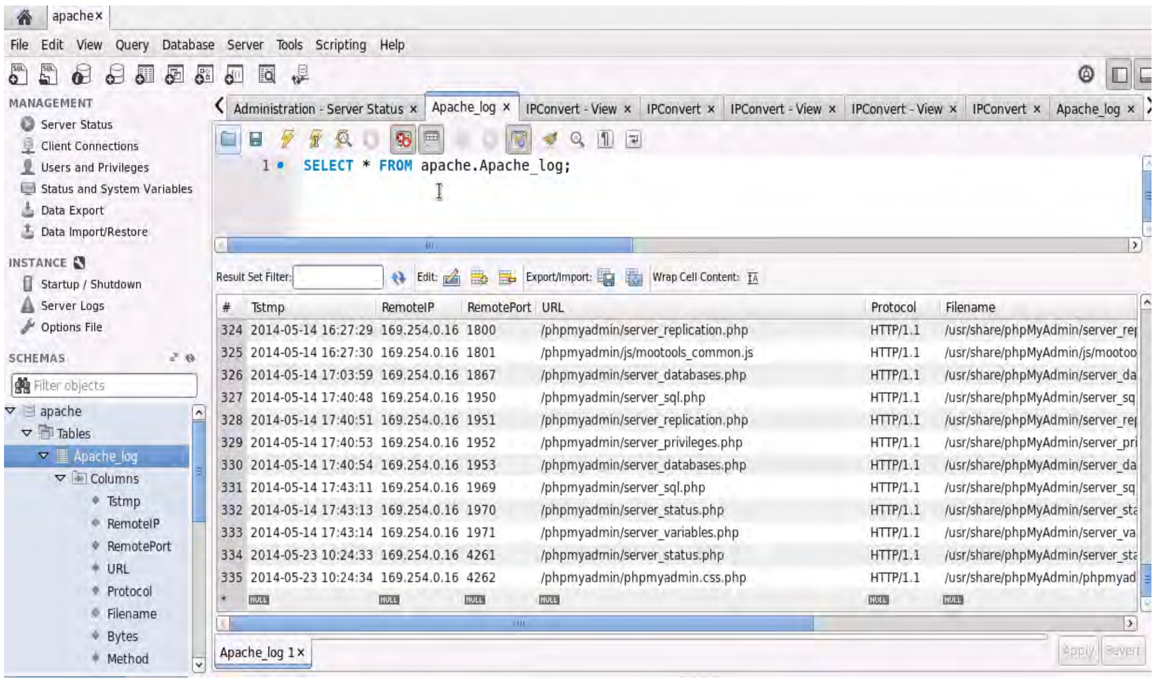
Figure 36. HTTPD configuration file showing the modification to include the Mod_Log_DBD parameters for writing to a database

4. **Restart Server then Start Services:**

- service mysqld start
- service httpd start

5. **Test the Database by Opening a Browser Window to URL http://127.0.0.1/ phpMyAdmin**

The database will immediately populate as shown below; see Figure 37.



```
SELECT * FROM apache.Apache_log;
```

#	Tstmp	RemoteIP	RemotePort	URL	Protocol	Filename
324	2014-05-14 16:27:29	169.254.0.16	1800	/phpmyadmin/server_replication.php	HTTP/1.1	/usr/share/phpMyAdmin/server_re
325	2014-05-14 16:27:30	169.254.0.16	1801	/phpmyadmin/js/mootools_common.js	HTTP/1.1	/usr/share/phpMyAdmin/js/mootoo
326	2014-05-14 17:03:59	169.254.0.16	1867	/phpmyadmin/server_databases.php	HTTP/1.1	/usr/share/phpMyAdmin/server_da
327	2014-05-14 17:40:48	169.254.0.16	1950	/phpmyadmin/server_sql.php	HTTP/1.1	/usr/share/phpMyAdmin/server_sq
328	2014-05-14 17:40:51	169.254.0.16	1951	/phpmyadmin/server_replication.php	HTTP/1.1	/usr/share/phpMyAdmin/server_re
329	2014-05-14 17:40:53	169.254.0.16	1952	/phpmyadmin/server_privileges.php	HTTP/1.1	/usr/share/phpMyAdmin/server_pri
330	2014-05-14 17:40:54	169.254.0.16	1953	/phpmyadmin/server_databases.php	HTTP/1.1	/usr/share/phpMyAdmin/server_da
331	2014-05-14 17:43:11	169.254.0.16	1969	/phpmyadmin/server_sql.php	HTTP/1.1	/usr/share/phpMyAdmin/server_sq
332	2014-05-14 17:43:13	169.254.0.16	1970	/phpmyadmin/server_status.php	HTTP/1.1	/usr/share/phpMyAdmin/server_sta
333	2014-05-14 17:43:14	169.254.0.16	1971	/phpmyadmin/server_variables.php	HTTP/1.1	/usr/share/phpMyAdmin/server_va
334	2014-05-23 10:24:33	169.254.0.16	4261	/phpmyadmin/server_status.php	HTTP/1.1	/usr/share/phpMyAdmin/server_sta
335	2014-05-23 10:24:34	169.254.0.16	4262	/phpmyadmin/phpmyadmin.css.php	HTTP/1.1	/usr/share/phpMyAdmin/phpmyad
*	NULL	NULL	NULL	NULL	NULL	NULL

Figure 37. A MySQL database using MySQL-Workbench as a GUI. Logs are parsed from the httpd subsystem by mod-dbd and written to the database with the mod-apr module.

B. GINA INSTALL

GINA consists of three core components: IIS front end, DotNetNuke environment, and MS SQL server. Aggregation of detection and response models require the GINA application, ODBC drivers, and Plink.

1. Standard GINA IIS Server

- IIS-configuration
- Dotenet Nuke configuration
- GINA configuration

2. Standard GINA SQL Server

- Installation
- Database restore
- Database permissions

3. ODBC Driver

In order for GINA to read in values from MySQL databases, a MySQL ODBC driver is required as well a special connection in GINA. The ODBC driver is installed on the GINA SQL server.

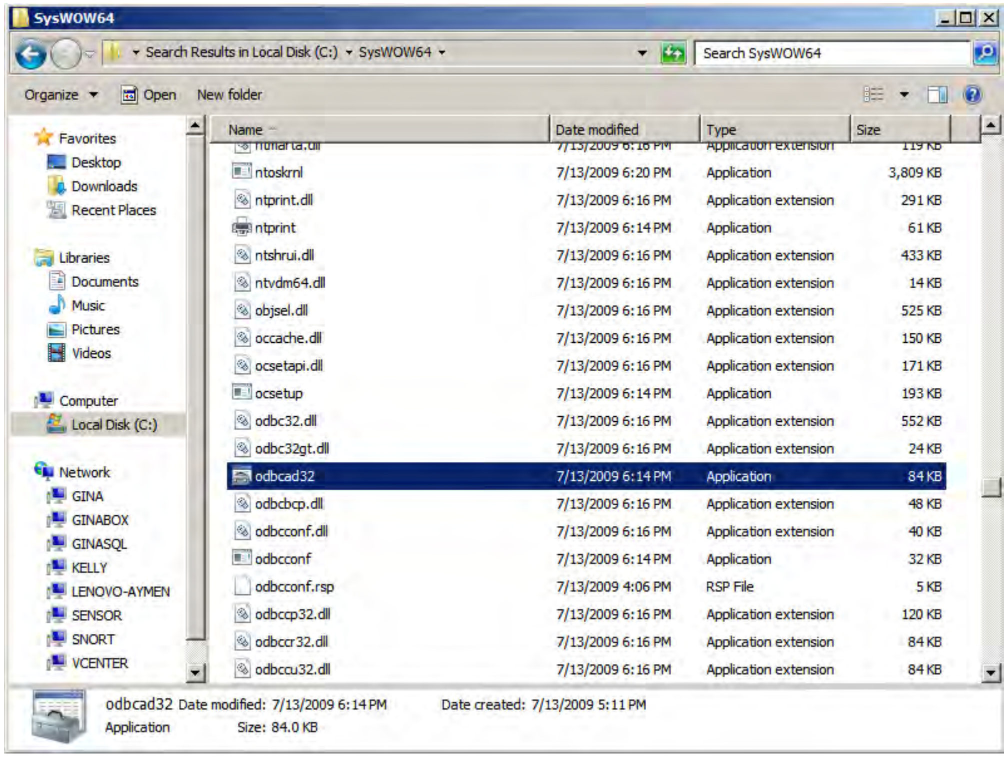
Procedure:

(1) Obtain the MySQL ODBC driver from http://dev.mysql.com/downloads/connector/odbc/ and install it on the GINA backend SQL server.

(2) *NOTE: We use the 32 bit program for a 32 bit install of GINA (this is the IIS. Even though we are using a 64 bit Linux MySQL server, 32 bit is the correct choice. 64 bit will not work on 32 bit GINA installs.

(3) Run the ODBC program c:\Windows\SysWOW64\ODBCAD32.exe; see Figure 38.

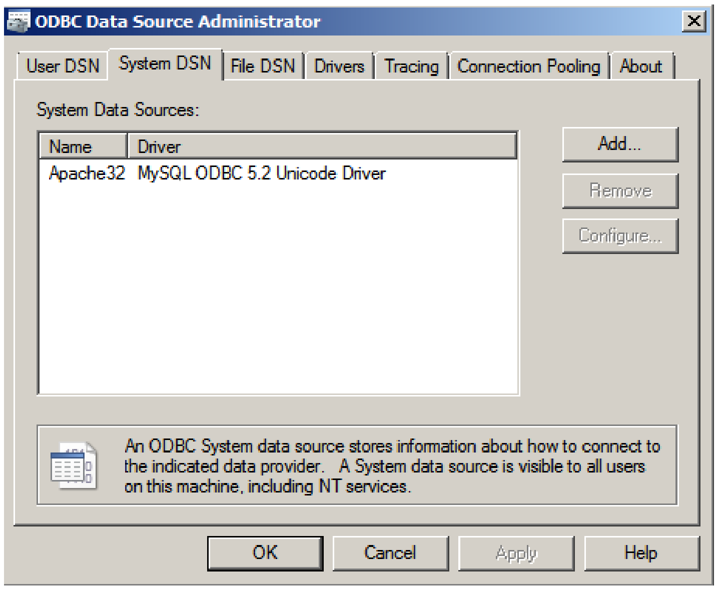
(4) Click the system DSN tab; see Figure 39.



Name	Date modified	Type	Size
ntmarta.dll	7/13/2009 6:16 PM	Application extension	119 KB
ntoskrnl	7/13/2009 6:20 PM	Application	3,809 KB
ntprint.dll	7/13/2009 6:16 PM	Application extension	291 KB
ntprint	7/13/2009 6:14 PM	Application	61 KB
ntshrui.dll	7/13/2009 6:16 PM	Application extension	433 KB
ntvdm64.dll	7/13/2009 6:16 PM	Application extension	14 KB
objsel.dll	7/13/2009 6:16 PM	Application extension	525 KB
occache.dll	7/13/2009 6:16 PM	Application extension	150 KB
ocsetapi.dll	7/13/2009 6:16 PM	Application extension	171 KB
ocsetup	7/13/2009 6:14 PM	Application	193 KB
odbc32.dll	7/13/2009 6:16 PM	Application extension	552 KB
odbc32gt.dll	7/13/2009 6:16 PM	Application extension	24 KB
odbcad32	7/13/2009 6:14 PM	Application	84 KB
odbcbcp.dll	7/13/2009 6:16 PM	Application extension	48 KB
odbcconf.dll	7/13/2009 6:16 PM	Application extension	40 KB
odbcconf	7/13/2009 6:14 PM	Application	32 KB
odbcconf.rsp	7/13/2009 4:06 PM	RSP File	5 KB
odbccp32.dll	7/13/2009 6:16 PM	Application extension	120 KB
odbccr32.dll	7/13/2009 6:16 PM	Application extension	84 KB
odbccu32.dll	7/13/2009 6:16 PM	Application extension	84 KB

odbcad32 Date modified: 7/13/2009 6:14 PM Date created: 7/13/2009 5:11 PM
Application Size: 84.0 KB

Figure 38. The ODBC driver configuration program is located on the GINA IIS server and is configured to communicate with a remote Linux MySQL server. This program is the link between MS SQL and MySQL databases.



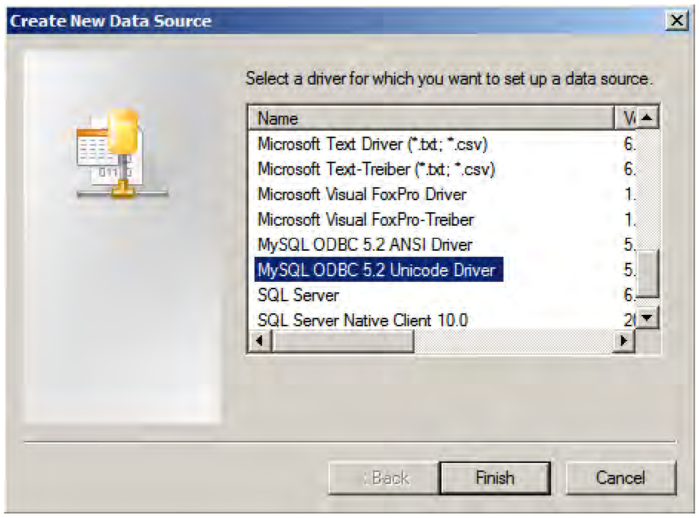
System Data Sources:

Name	Driver
Apache32	MySQL ODBC 5.2 Unicode Driver

An ODBC System data source stores information about how to connect to the indicated data provider. A System data source is visible to all users on this machine, including NT services.

Figure 39. ODBC configurations are indexed in this GUI. A connection for each MySQL server must exist in this data source with a name. The name is referenced as an X-Type Source in VRDM.

(5) Click add and select MySQL ODBC 5.2 Unicode Driver; see Figure 40.



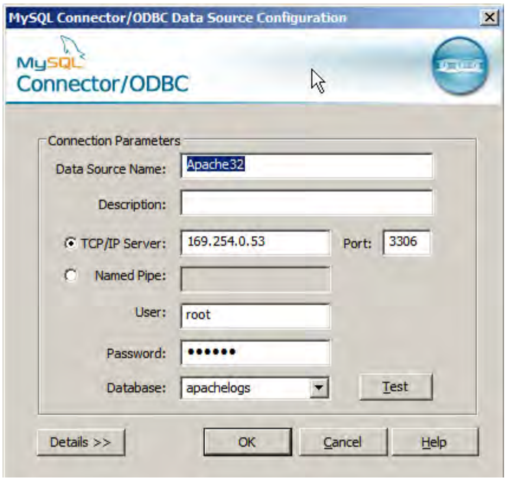
Create New Data Source

Select a driver for which you want to set up a data source.

Name	V
Microsoft Text Driver (*.txt; *.csv)	6.
Microsoft Text-Treiber (*.txt; *.csv)	6.
Microsoft Visual FoxPro Driver	1.
Microsoft Visual FoxPro-Treiber	1.
MySQL ODBC 5.2 ANSI Driver	5.
MySQL ODBC 5.2 Unicode Driver	5.
SQL Server	6.
SQL Server Native Client 10.0	2(

Figure 40. Many drivers can be loaded in the ODBC administration tool. This tool is a middleware component that is transparent once used as a source to VRDM X-Types.

(6) Parameters required are: Data source name (This is what is referenced in GINA in the sources window), TCP/IP server, Port, Username, Password, and the database name on the remote server; see Figure 41.



MySQL Connector/ODBC Data Source Configuration

MySQL Connector/ODBC

Connection Parameters

Data Source Name: Apache32

Description:

(●) TCP/IP Server: 169.254.0.53 Port: 3306

( ) Named Pipe:

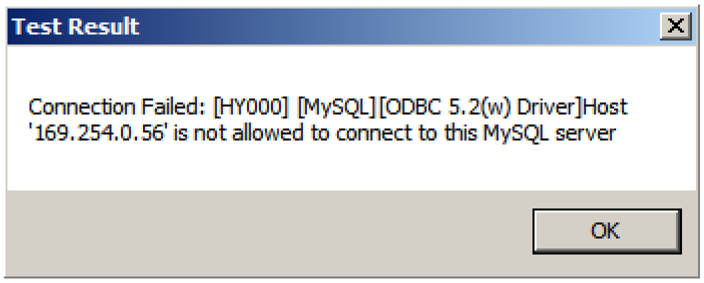
User: root

Password: ••••••

Database: apachelogs

Figure 41. The MySQL ODBC driver contains the IP address and port of the remote MySQL server. GINA references the “Data Source Name” as an X-Type Source.

(7) When you click test you may get this; see Figure 42.



Test Result

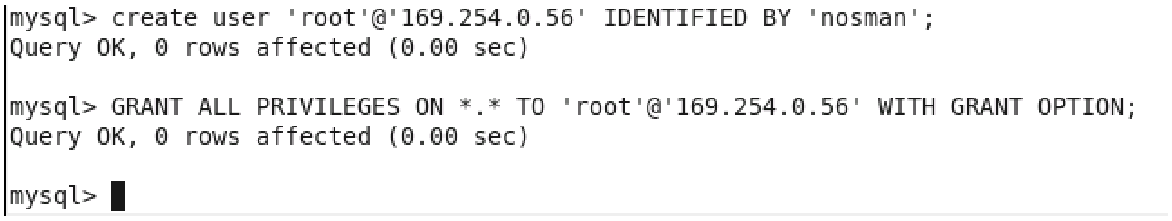
Connection Failed: [HY000] [MySQL][ODBC 5.2(w) Driver]Host '169.254.0.56' is not allowed to connect to this MySQL server

Figure 42. Authentication errors are common in system integrations. Errors like this can be mitigated by ensuring credentials exist on the remote data source before configuring the local connection.

4. Apply Permissions to SQL Server for Remote Access

Procedure:

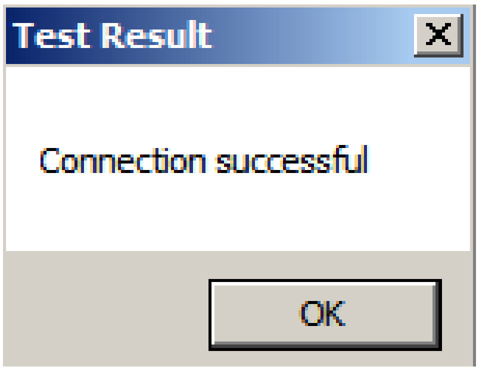
- mysql –u root –p<password here with no space after the -p>
- CREATE USER '<username>'@'<gina server IP address>' IDENTIFIED BY 'password';
- GRANT ALL PRIVILEGES ON *.* TO 'root'@'169.254.0.59' WHICH GRANT OPTION; see Figure 44.



```
mysql> create user 'root'@'169.254.0.56' IDENTIFIED BY 'nosman';
Query OK, 0 rows affected (0.00 sec)

mysql> GRANT ALL PRIVILEGES ON *.* TO 'root'@'169.254.0.56' WITH GRANT OPTION;
Query OK, 0 rows affected (0.00 sec)

mysql> 
```

Figure 43. MySQL Command Reference

- Attempt a test in the ODBC connector configuration [Test] button; see Figure 44.



Connection successful

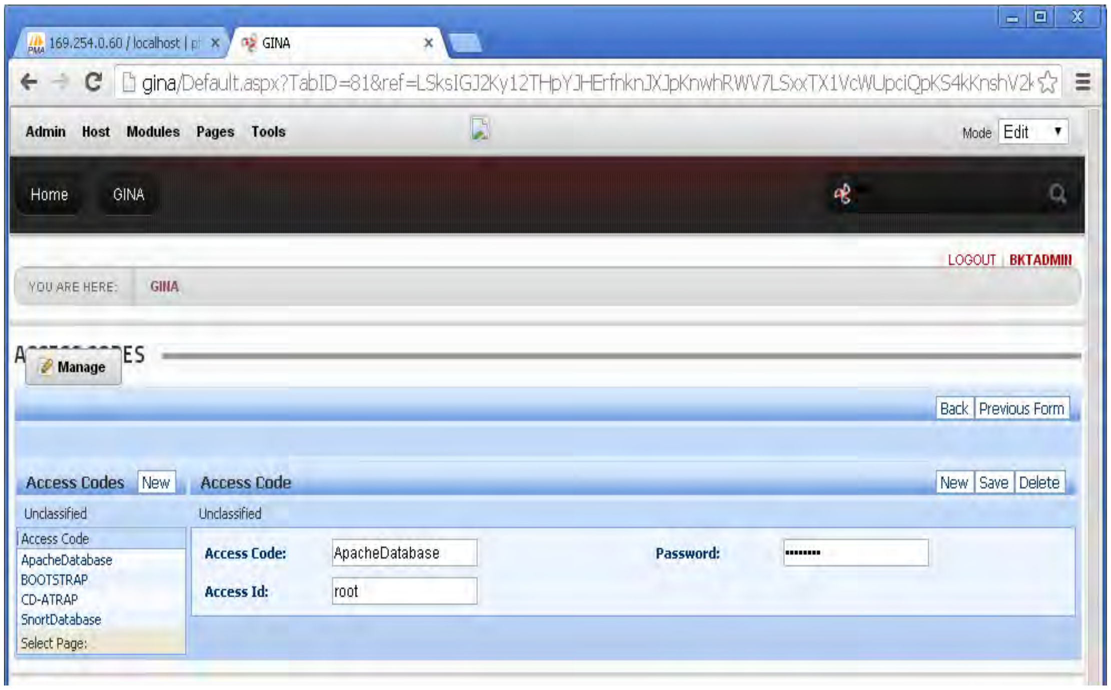
Figure 44. The MySQL driver has an ODBC test option that verifies connectivity from the IIS server to the MySQL database server. Connectivity to the MySQL driver is verified from within VRDM by creating a form to verify the MySQL data can hydrate a form field.

5. A GINA Source for the ODBC Connector Is Required

Procedure:

- Create an access code:

GINA/GINA/Users/Access Codes/New; see Figure 45.

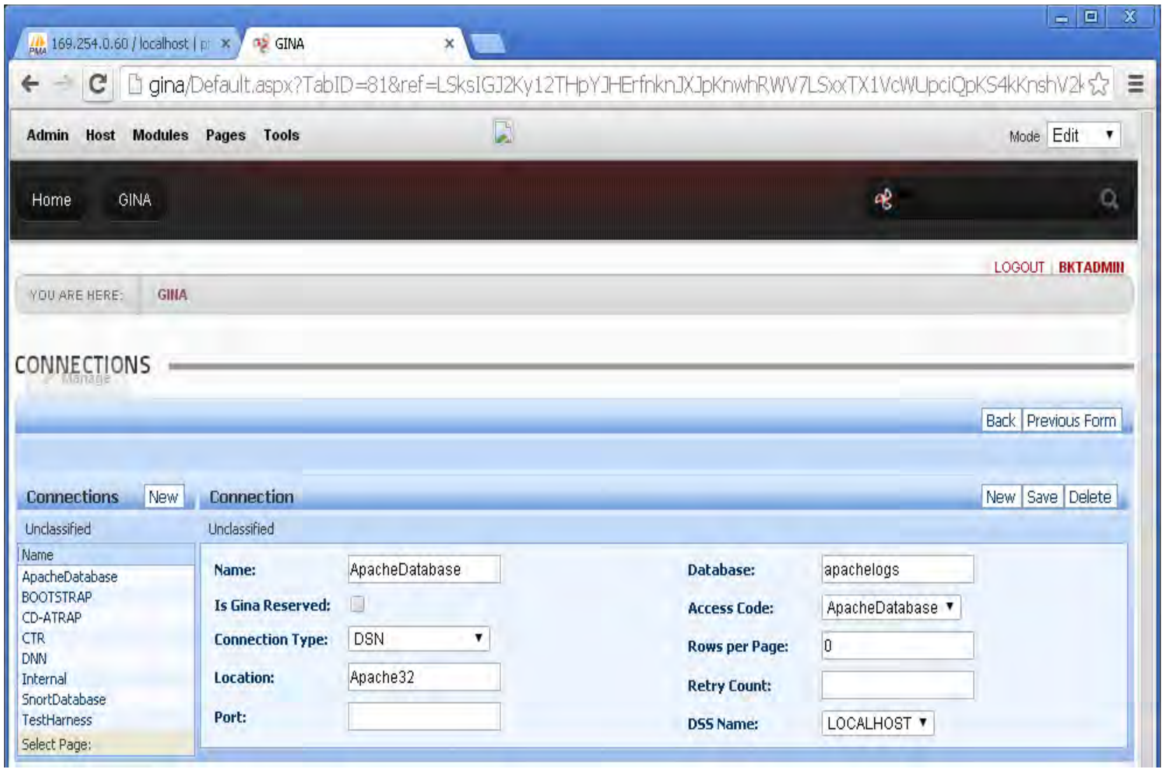


Access Code: ApacheDatabase
Password: ••••••••
Access Id: root

Figure 45. Database access codes are configured as a VRDM object and saved in a MS SQL server. MS SQL supports secure socket layer (SSL) and Internet Protocol Security (IPSec) to encrypt the database credentials in transit.

- Create a connection that has the same name as the ODBC connection created above.

GINA/GINA/Sources/Connections/New; see Figure 46.

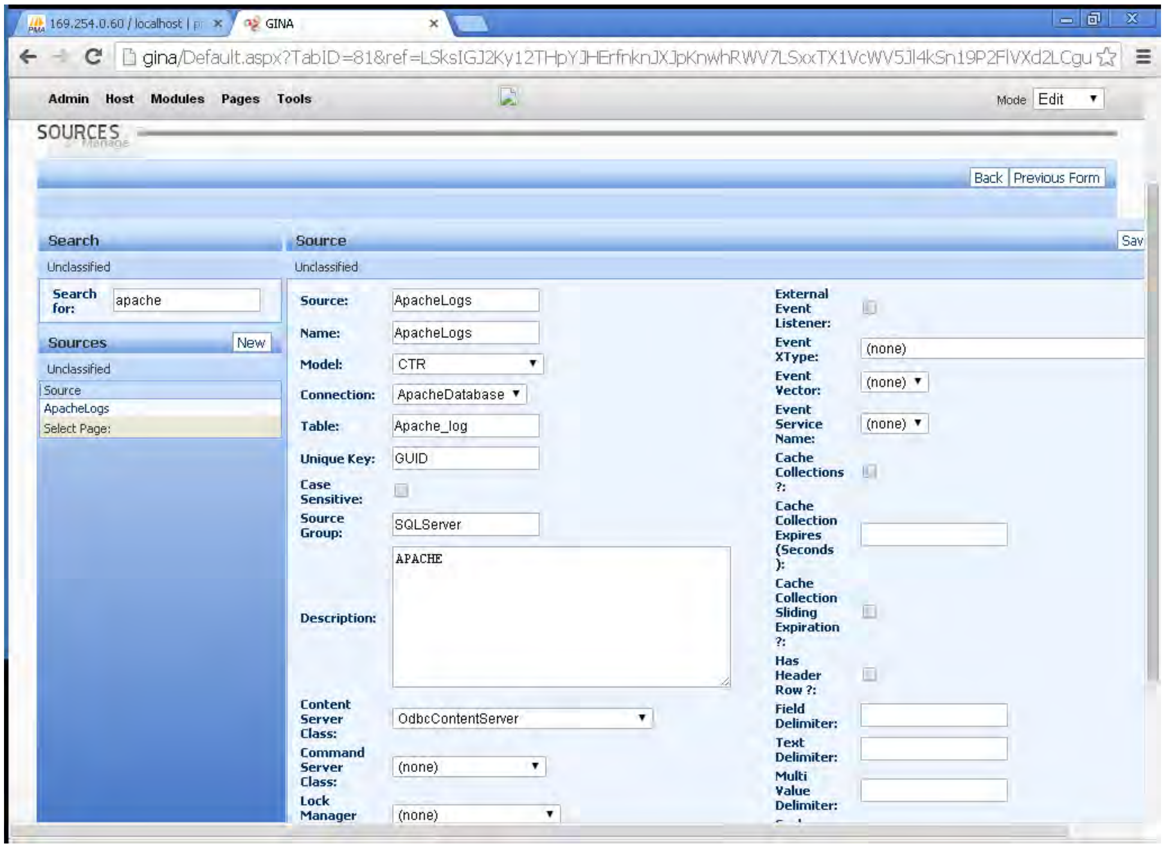


Name:	ApacheDatabase	Database:	apachelogs
Is Gina Reserved:	☐	Access Code:	ApacheDatabase
Connection Type:	DSN	Rows per Page:	0
Location:	Apache32	Retry Count:	
Port:		DSS Name:	LOCALHOST

Figure 46. ODBC connections are defined in the same manner as a database source. The location is the ODBC driver connection name instead of the MySQL database name. In essence, this connection is local to the ODBC driver loaded on the integrated IIS server.

- Create the source

GINA/GINA/Sources NEW; see Figure 47.


169.254.0.60 / localhost | p
GINA
gina/Default.aspx?TabID=81&ref=LSksIGJ2Ky12THpYJHErfnknJXJpKnwhRWV7LSxxTX1VcWV5Jl4kSn19P2FlVXd2LCgu
Admin
Host
Modules
Pages
Tools
Mode
SOURCES
Manage
Back
Previous Form
Search
Source
Sav
Unclassified
New
Sources
Select Page:

Search for:	apache
Sources	Unclassified / Source / ApacheLogs

Source:	ApacheLogs
Name:	ApacheLogs
Model:	CTR
Connection:	ApacheDatabase
Table:	Apache_log
Unique Key:	GUID
Case Sensitive:	☐
Source Group:	SQLServer
Description:	APACHE
Content Server Class:	OdbcContentServer
Command Server Class:	(none)
Lock Manager	(none)
External Event Listener:	☐
Event XType:	(none)
Event Vector:	(none)
Event Service Name:	(none)
Cache Collections ?:	☐
Cache Collection Expires (Seconds):	
Cache Collection Sliding Expiration ?:	☐
Has Header Row ?:	☐
Field Delimiter:	
Text Delimiter:	
Multi Value Delimiter:	

Figure 47. A Source can be thought of as a valid table name which has been given a nickname. Here, the table name “Apache_log” is named “ApacheLogs.”

The source can be referenced by X-Types inside the GINA modeling environment.

6. Install Plink

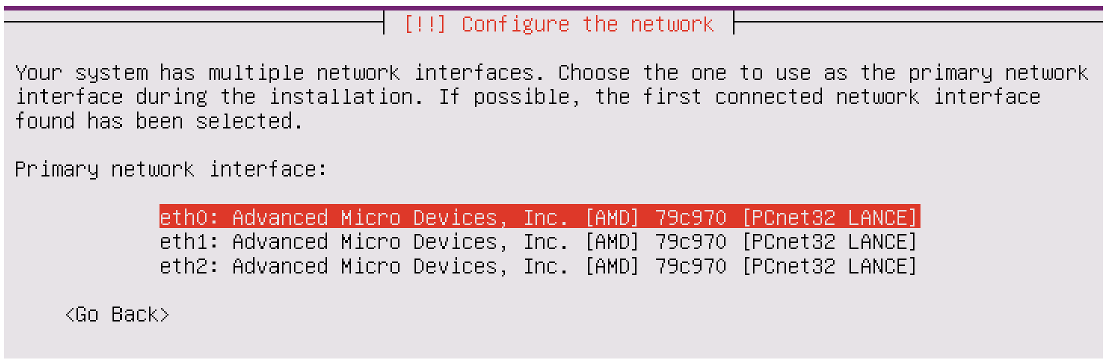
Procedure

- Plink can be downloaded from http://the.earth.li/~sgtatham/putty/latest/x86/plink.exe. It is a small and simple program that does not require complicated installation procedures.

7. Install Ubuntu as Router with IPTables

Procedure

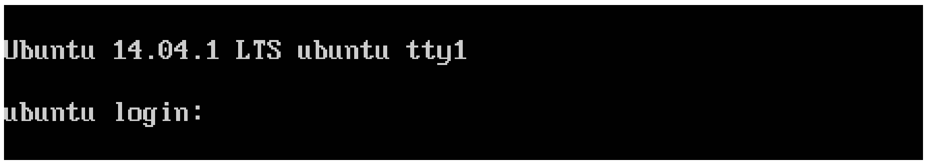
- Upon boot the system will ask for a user name and password before continuing
- Enter user and password and make note as these will be the primary login for the system.
- Initial install of Ubuntu will which of your NICs will be primary for the system (typically eth 0) please note it will be needed later during the install; see Figure 48.



```
[!!] Configure the network

Your system has multiple network interfaces. Choose the one to use as the primary network
interface during the installation. If possible, the first connected network interface
found has been selected.

Primary network interface:

            eth0: Advanced Micro Devices, Inc. [AMD] 79c970 [PCnet32 LANCE]
            eth1: Advanced Micro Devices, Inc. [AMD] 79c970 [PCnet32 LANCE]
            eth2: Advanced Micro Devices, Inc. [AMD] 79c970 [PCnet32 LANCE]

    <Go Back>
```

Figure 48. Primary NIC Selection.

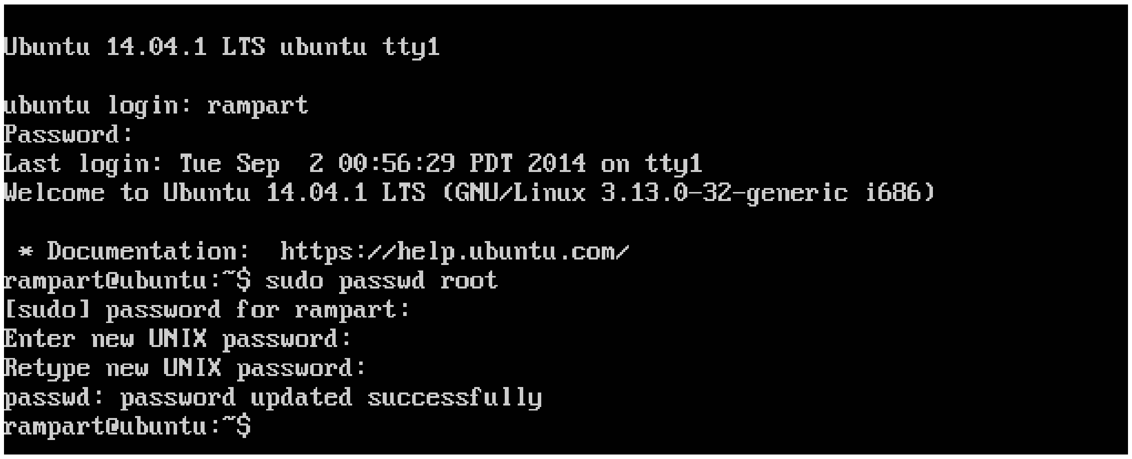
- The system will install automatically and reboot to the login screen. Login to the system using aforementioned credentials; see Figure 49.



```
Ubuntu 14.04.1 LTS ubuntu tty1

ubuntu login:
```

Figure 49. Ubuntu Login Prompt.

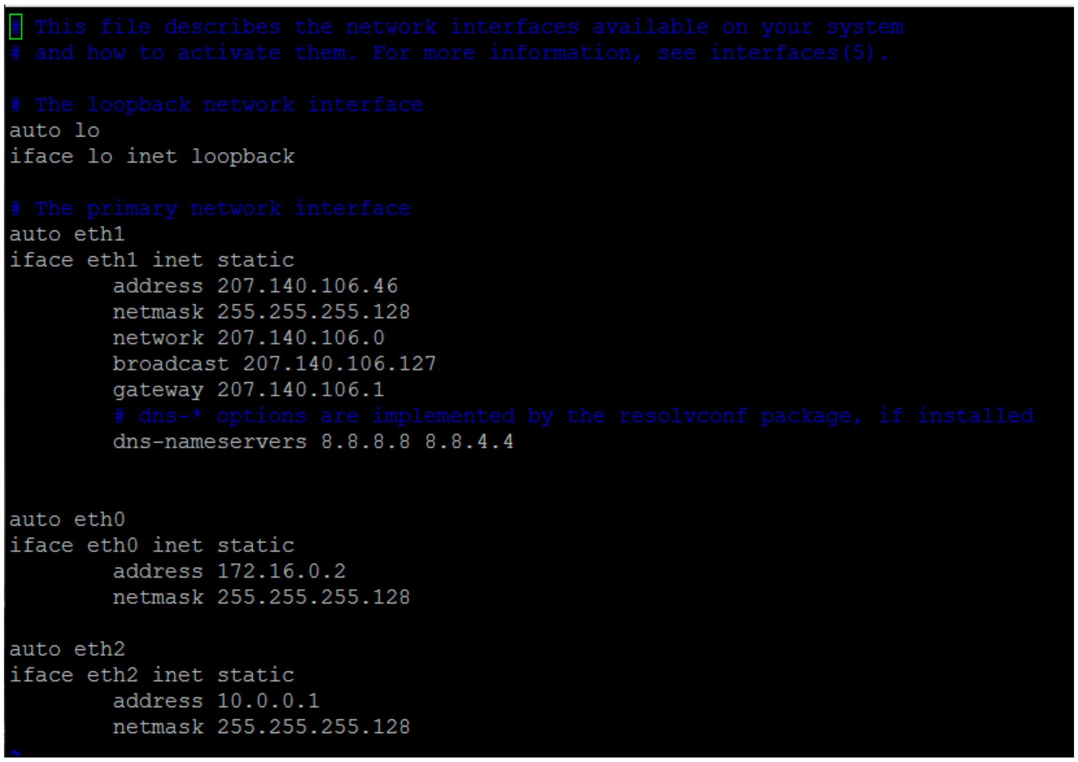
- By default the root user is disabled. To enable the root account type the following and set a password.
- sudo passwd root, then enter and password for the root account; see Figure 50.



```
Ubuntu 14.04.1 LTS ubuntu tty1

ubuntu login: rampart
Password:
Last login: Tue Sep  2 00:56:29 PDT 2014 on tty1
Welcome to Ubuntu 14.04.1 LTS (GNU/Linux 3.13.0-32-generic i686)

 * Documentation:  https://help.ubuntu.com/
rampart@ubuntu:~$ sudo passwd root
[sudo] password for rampart:
Enter new UNIX password:
Retype new UNIX password:
passwd: password updated successfully
rampart@ubuntu:~$
```

Figure 50. Enabling Root Account.

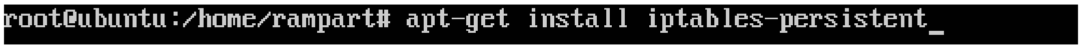
- Set the network interface IP's by navigating to the c/etc/network folder then vi the interfaces file; see Figure 51.



```
# This file describes the network interfaces available on your system
# and how to activate them. For more information, see interfaces(5).

# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
auto eth1
iface eth1 inet static
        address 207.140.106.46
        netmask 255.255.255.128
        network 207.140.106.0
        broadcast 207.140.106.127
        gateway 207.140.106.1
        # dns-* options are implemented by the resolvconf package, if installed
        dns-nameservers 8.8.8.8 8.8.4.4


auto eth0
iface eth0 inet static
        address 172.16.0.2
        netmask 255.255.255.128

auto eth2
iface eth2 inet static
        address 10.0.0.1
        netmask 255.255.255.128
~
```

Figure 51. Sample Interfaces File.

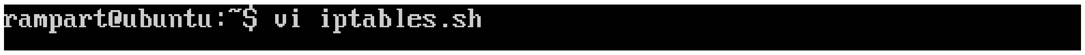
- Install the IPTables-persistent package to allow for reloading of tables if system restart is required; see Figure 52.



```
root@ubuntu:/home/rampart# apt-get install iptables-persistent_
```

Figure 52. Installing IPTables-Persistent Package.

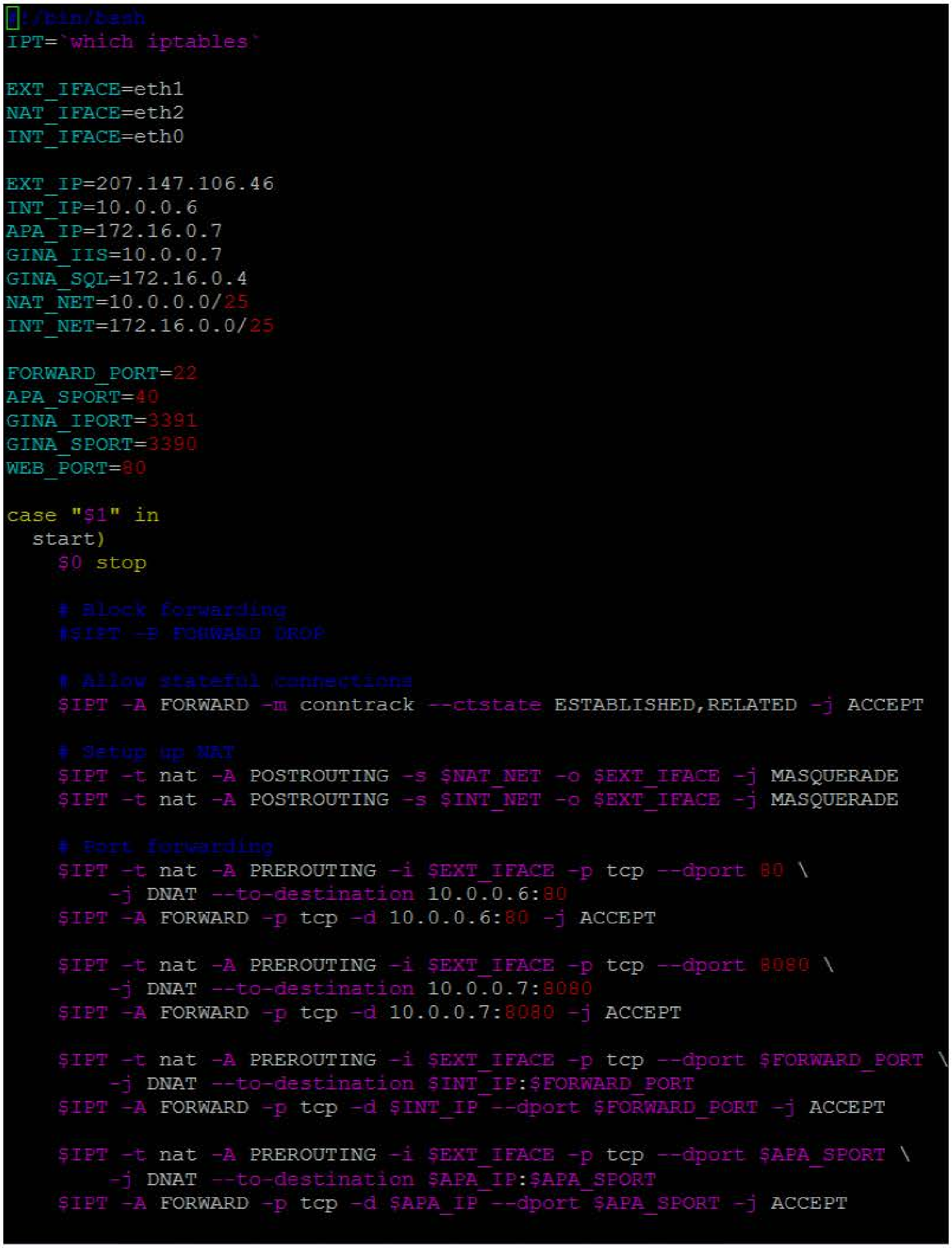
- Prepare your IPTables file by using vi to create a file to build the IPTables from; see Figure 53.



```
rampart@ubuntu:~$ vi iptables.sh
```

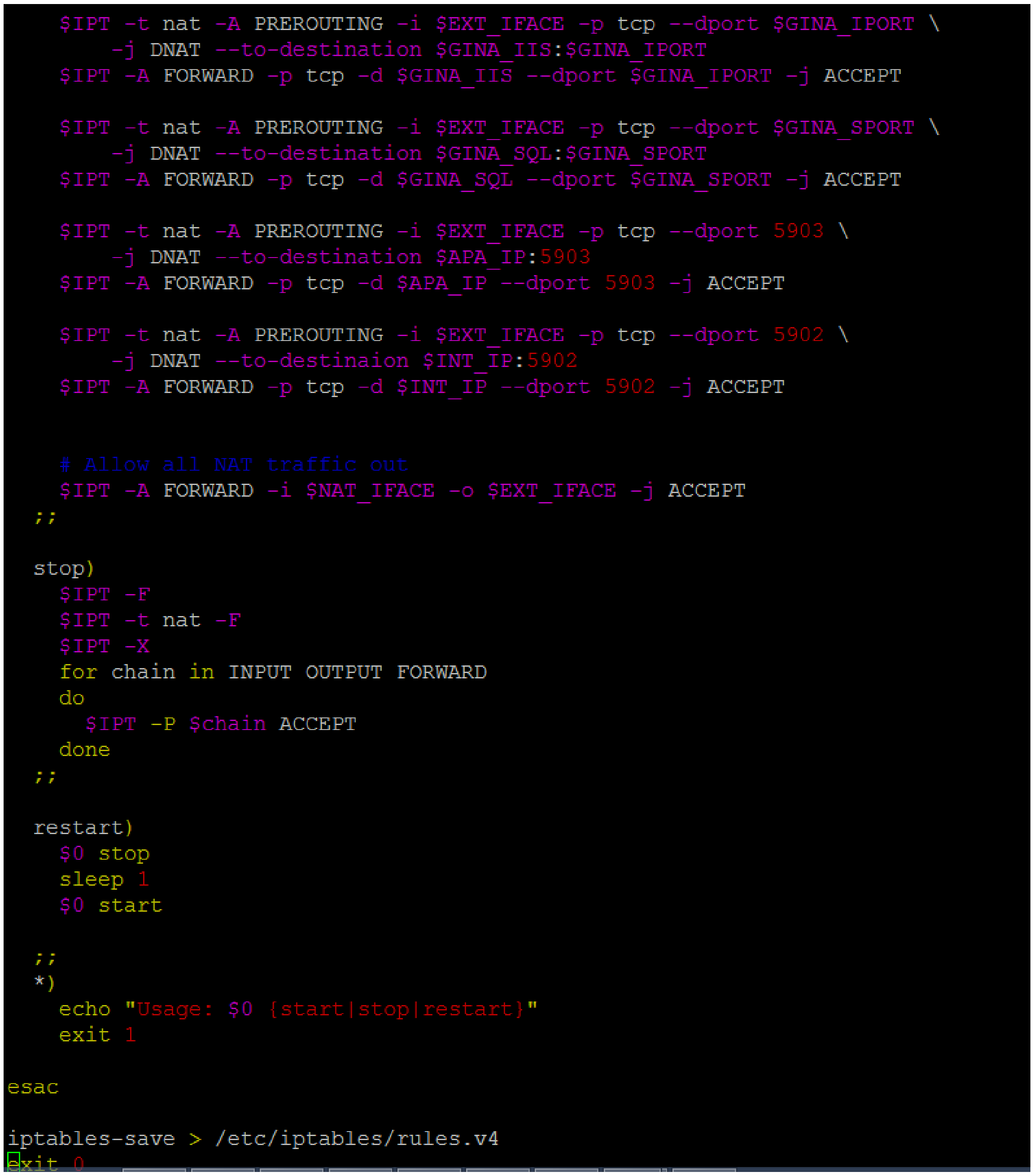
Figure 53. Creating IPTables File.

- To allow for ease of updating the IPTables build a small script to load the IPTables from; see Figures 54 and 55.



```
#!/bin/bash
IPT=`which iptables`

EXT_IFACE=eth1
NAT_IFACE=eth2
INT_IFACE=eth0

EXT_IP=207.147.106.46
INT_IP=10.0.0.6
APA_IP=172.16.0.7
GINA_IIS=10.0.0.7
GINA_SQL=172.16.0.4
NAT_NET=10.0.0.0/25
INT_NET=172.16.0.0/25

FORWARD_PORT=22
APA_SPORT=40
GINA_IPORT=3391
GINA_SPORT=3390
WEB_PORT=80

case "$1" in
  start)
    $0 stop

    # Block forwarding
    #$IPT -P FORWARD DROP

    # Allow stateful connections
    $IPT -A FORWARD -m conntrack --ctstate ESTABLISHED,RELATED -j ACCEPT

    # Setup up NAT
    $IPT -t nat -A POSTROUTING -s $NAT_NET -o $EXT_IFACE -j MASQUERADE
    $IPT -t nat -A POSTROUTING -s $INT_NET -o $EXT_IFACE -j MASQUERADE

    # Port forwarding
    $IPT -t nat -A PREROUTING -i $EXT_IFACE -p tcp --dport 80 \
        -j DNAT --to-destination 10.0.0.6:80
    $IPT -A FORWARD -p tcp -d 10.0.0.6:80 -j ACCEPT

    $IPT -t nat -A PREROUTING -i $EXT_IFACE -p tcp --dport 8080 \
        -j DNAT --to-destination 10.0.0.7:8080
    $IPT -A FORWARD -p tcp -d 10.0.0.7:8080 -j ACCEPT

    $IPT -t nat -A PREROUTING -i $EXT_IFACE -p tcp --dport $FORWARD_PORT \
        -j DNAT --to-destination $INT_IP:$FORWARD_PORT
    $IPT -A FORWARD -p tcp -d $INT_IP --dport $FORWARD_PORT -j ACCEPT

    $IPT -t nat -A PREROUTING -i $EXT_IFACE -p tcp --dport $APA_SPORT \
        -j DNAT --to-destination $APA_IP:$APA_SPORT
    $IPT -A FORWARD -p tcp -d $APA_IP --dport $APA_SPORT -j ACCEPT
```

Figure 54. Sample IPTables File (1 of 2).



```
    $IPT -t nat -A PREROUTING -i $EXT_IFACE -p tcp --dport $GINA_IPORT \
        -j DNAT --to-destination $GINA_IIS:$GINA_IPORT
    $IPT -A FORWARD -p tcp -d $GINA_IIS --dport $GINA_IPORT -j ACCEPT

    $IPT -t nat -A PREROUTING -i $EXT_IFACE -p tcp --dport $GINA_SPORT \
        -j DNAT --to-destination $GINA_SQL:$GINA_SPORT
    $IPT -A FORWARD -p tcp -d $GINA_SQL --dport $GINA_SPORT -j ACCEPT

    $IPT -t nat -A PREROUTING -i $EXT_IFACE -p tcp --dport 5903 \
        -j DNAT --to-destination $APA_IP:5903
    $IPT -A FORWARD -p tcp -d $APA_IP --dport 5903 -j ACCEPT

    $IPT -t nat -A PREROUTING -i $EXT_IFACE -p tcp --dport 5902 \
        -j DNAT --to-destinaion $INT_IP:5902
    $IPT -A FORWARD -p tcp -d $INT_IP --dport 5902 -j ACCEPT


    # Allow all NAT traffic out
    $IPT -A FORWARD -i $NAT_IFACE -o $EXT_IFACE -j ACCEPT
  ;;

  stop)
    $IPT -F
    $IPT -t nat -F
    $IPT -X
    for chain in INPUT OUTPUT FORWARD
    do
      $IPT -P $chain ACCEPT
    done
  ;;

  restart)
    $0 stop
    sleep 1
    $0 start

  ;;
  *)
    echo "Usage: $0 {start|stop|restart}"
    exit 1

esac

iptables-save > /etc/iptables/rules.v4
exit 0
```

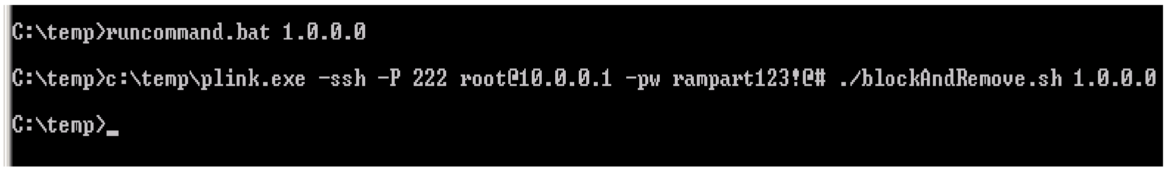
Figure 55. Sample IPTables File (2 of 2).

- Save the IPTables to the rules.v4 to make them persistent by typing "iptables-save > /etc/iptables/rules.v4" without the quotes then reboot the system and check for proper routing of traffic based on the new rules.

THIS PAGE INTENTIONALLY LEFT BLANK

APPENDIX G: ROUTER UPDATE SCRIPT FIXES

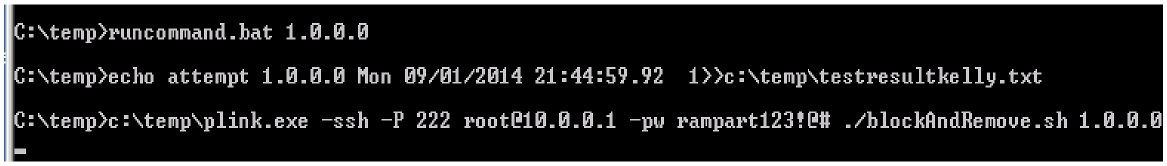
Team RAMPART tested the Responder module to ensure it was properly updating the router with threat IP addresses. The team started by running the block command manually from a command prompt on the GINA server. The command completed the IPTable update to the router and blocked the IP address from future attacks; see Figure 56.



```
C:\temp>runcommand.bat 1.0.0.0

C:\temp>c:\temp\plink.exe -ssh -P 222 root@10.0.0.1 -pw rampart123!@# ./blockAndRemove.sh 1.0.0.0

C:\temp>_
```

Figure 56. Manual execution of router update.

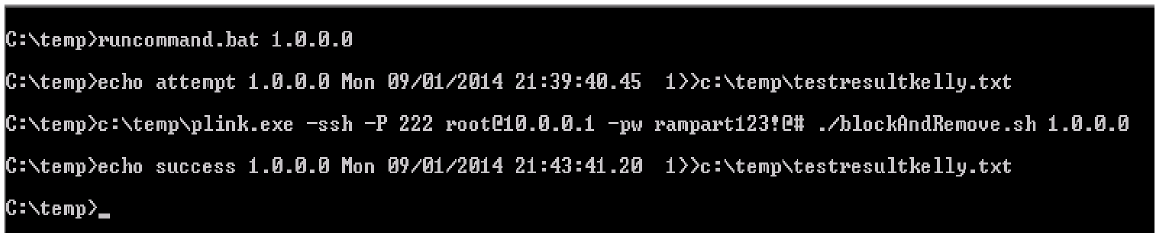
As the team transitioned from manually testing to fully automated testing of the system, it was discovered that the command was not completing; see Figure 57.



```
C:\temp>runcommand.bat 1.0.0.0

C:\temp>echo attempt 1.0.0.0 Mon 09/01/2014 21:44:59.92  1>>c:\temp\testresultkelly.txt

C:\temp>c:\temp\plink.exe -ssh -P 222 root@10.0.0.1 -pw rampart123!@# ./blockAndRemove.sh 1.0.0.0
_
```

Figure 57. Fully automated system fault.

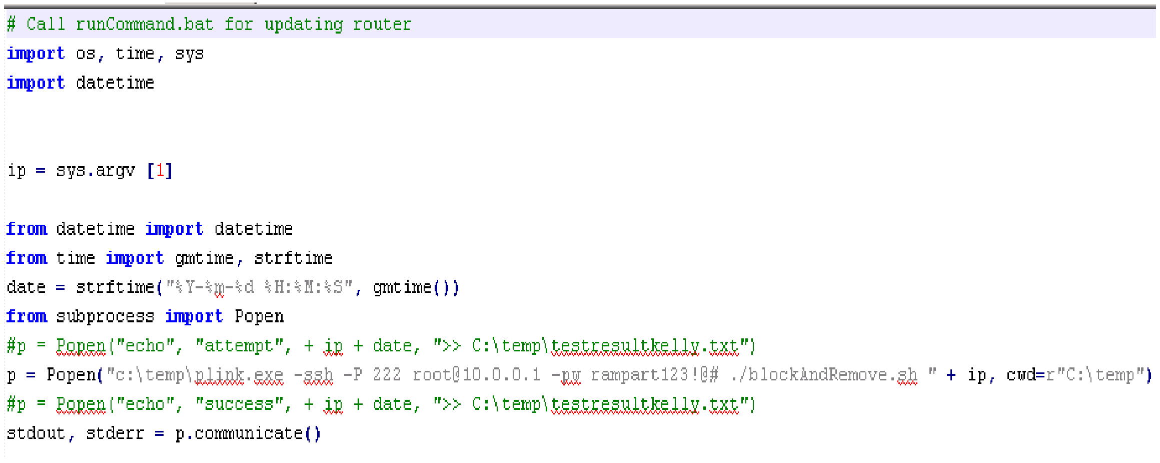
GINA was stopped by the router and not allowed to execute the IPTable update. The team investigated the cause of this problem by adding echo lines to the runcommand file in order to track its progress throughout execution. This procedure also allowed another form of logging with timestamps for future evaluation; see Figure 58.



```
C:\temp>runcommand.bat 1.0.0.0

C:\temp>echo attempt 1.0.0.0 Mon 09/01/2014 21:39:40.45  1>>c:\temp\testresultkelly.txt

C:\temp>c:\temp\plink.exe -ssh -P 222 root@10.0.0.1 -pw rampart123!@# ./blockAndRemove.sh 1.0.0.0

C:\temp>echo success 1.0.0.0 Mon 09/01/2014 21:43:41.20  1>>c:\temp\testresultkelly.txt

C:\temp>_
```

Figure 58. Echo lines added to the runcommand file.

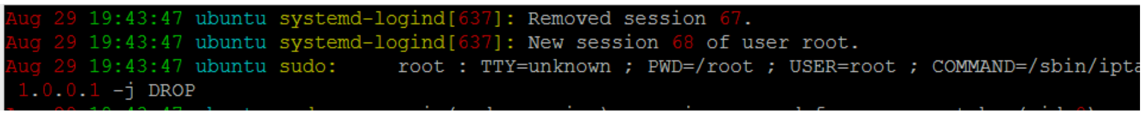
The completion of the batch file used in updating the router informed the team that it was a communication issue between the GINA server and the router. Based on the proper writing of the IP address and time stamps to the testresult.txts file by the script, the disconnect lay between the batch file and the router. In an attempt to isolate the problem, a Python script was written to test if the batch file was actually calling and executing the script; see Figure 59.



```
# Call runCommand.bat for updating router
import os, time, sys
import datetime


ip = sys.argv [1]

from datetime import datetime
from time import gmtime, strftime
date = strftime("%Y-%m-%d %H:%M:%S", gmtime())
from subprocess import Popen
#p = Popen("echo", "attempt", + ip + date, ">> C:\temp\testresultkelly.txt")
p = Popen("c:\temp\plink.exe -ssh -P 222 root@10.0.0.1 -pw rampart123!@# ./blockAndRemove.sh " + ip, cwd=r"C:\temp")
#p = Popen("echo", "success", + ip + date, ">> C:\temp\testresultkelly.txt")
stdout, stderr = p.communicate()
```

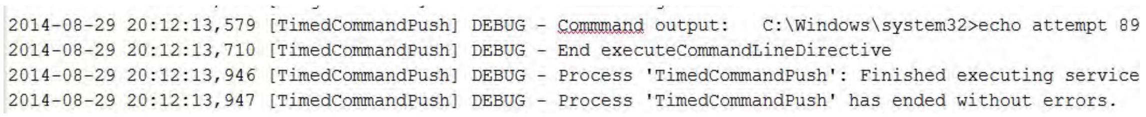
Figure 59. Python Script to Test the batch File.

The sole purpose of this file was to take the batch file out of the equation to assess if GINA was properly executing the runcommand. Again, the file executed and wrote the test IP addresses to the text files, however failed to update the router. Additionally, the team looked at different logs produced by programs in use on the GINA server to ensure there were no errors in accessing the batch file. Also, the auth.log file on the router was checked for access denials when the GINA service attempted to run the update. The team discovered that there were no access attempts in the auth.log when GINA called the process. A valid access was logged when the command was manually executed as identified below; see Figure 60, 61 and 62.



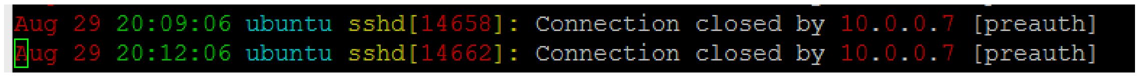
```
Aug 29 19:43:47 ubuntu systemd-logind[637]: Removed session 67.
Aug 29 19:43:47 ubuntu systemd-logind[637]: New session 68 of user root.
Aug 29 19:43:47 ubuntu sudo:     root : TTY=unknown ; PWD=/root ; USER=root ; COMMAND=/sbin/ipta
 1.0.0.1 -j DROP
```

Figure 60. Manual execution of the batch file to update the router.



```
2014-08-29 20:12:13,579 [TimedCommandPush] DEBUG - Commmand output:   C:\Windows\system32>echo attempt 89
2014-08-29 20:12:13,710 [TimedCommandPush] DEBUG - End executeCommandLineDirective
2014-08-29 20:12:13,946 [TimedCommandPush] DEBUG - Process 'TimedCommandPush': Finished executing service
2014-08-29 20:12:13,947 [TimedCommandPush] DEBUG - Process 'TimedCommandPush' has ended without errors.
```

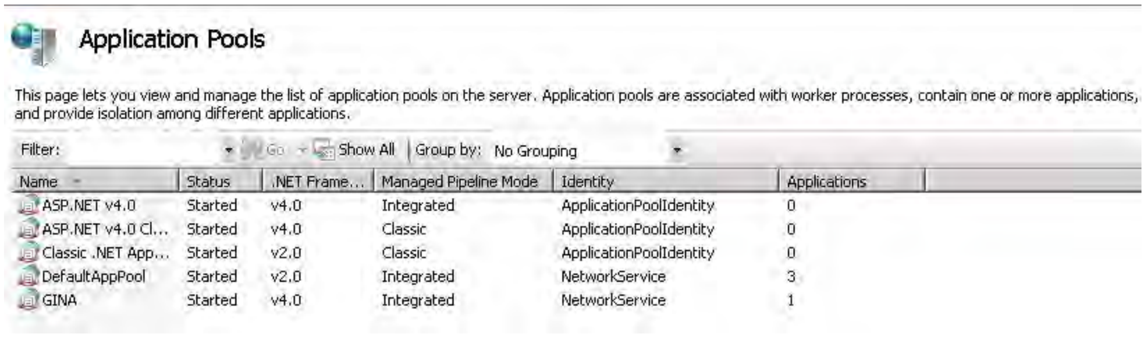
Figure 61. GINA attempt to remotely update the router.



```
Aug 29 20:09:06 ubuntu sshd[14658]: Connection closed by 10.0.0.7 [preauth]
Aug 29 20:12:06 ubuntu sshd[14662]: Connection closed by 10.0.0.7 [preauth]
```

Figure 62. Failed login by the automated script via GINA.

Next, the team looked at which user was executing the runcommand and the application pool of the IIS server. The team found that the user was a not a normal user account. The next change to execution was made to the application pool user for both the default and GINA application pools. The default user for this web service was “network service”; see Figure 63.



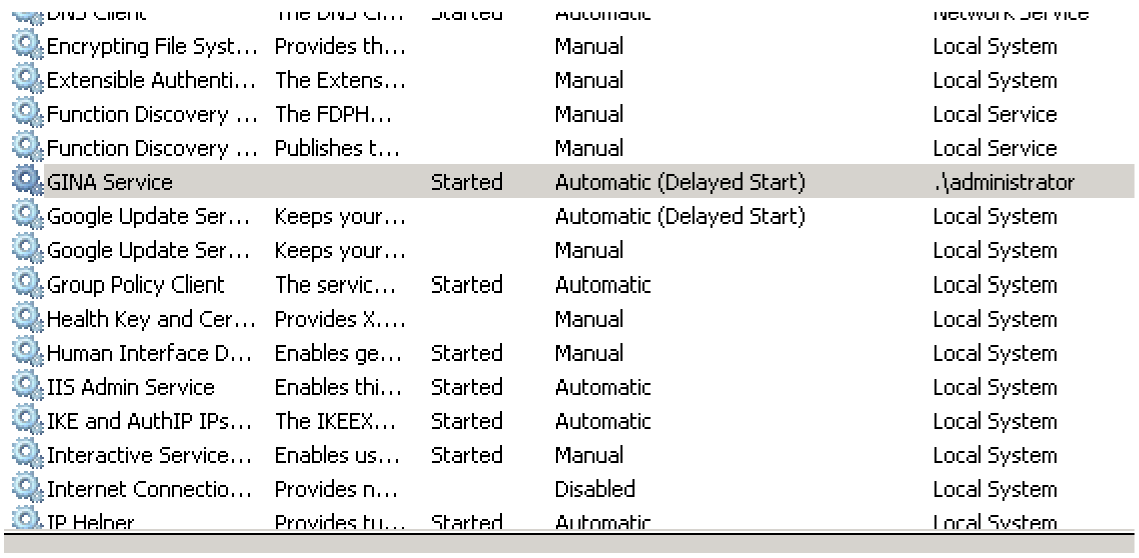
Application Pools

This page lets you view and manage the list of application pools on the server. Application pools are associated with worker processes, contain one or more applications, and provide isolation among different applications.

Name	Status	.NET Frame...	Managed Pipeline Mode	Identity	Applications
ASP.NET v4.0	Started	v4.0	Integrated	ApplicationPoolIdentity	0
ASP.NET v4.0 Cl...	Started	v4.0	Classic	ApplicationPoolIdentity	0
Classic .NET App...	Started	v2.0	Classic	ApplicationPoolIdentity	0
DefaultAppPool	Started	v2.0	Integrated	NetworkService	3
GINA	Started	v4.0	Integrated	NetworkService	1

Figure 63. GINA IIS Application Pools.

Modification of the user in the IIS server itself did not result in updating the router. Also, the team learned that the GINA service is what controls the communication between GINA and the router. This service or user account is what the router sees each time a connection is attempted. With respect to SSH, the first time a user connects to a server they must accept the key from the server. The team’s inability to physically log onto the server as the user localsystem and SSH to the router to accept the key posed a problem. After speaking with a BKT technician about which users the GINA service was authorized to run, the team switched from the localsystem account to the administrator account; see Figure 64.



DNS Client	The DNS Cl…	Started	Automatic	Network Service
Encrypting File Syst…	Provides th…		Manual	Local System
Extensible Authenti…	The Extens…		Manual	Local System
Function Discovery …	The FDPH…		Manual	Local Service
Function Discovery …	Publishes t…		Manual	Local Service
GINA Service		Started	Automatic (Delayed Start)	.\administrator
Google Update Ser…	Keeps your…		Automatic (Delayed Start)	Local System
Google Update Ser…	Keeps your…		Manual	Local System
Group Policy Client	The servic…	Started	Automatic	Local System
Health Key and Cer…	Provides X.…		Manual	Local System
Human Interface D…	Enables ge…	Started	Manual	Local System
IIS Admin Service	Enables thi…	Started	Automatic	Local System
IKE and AuthIP IPs…	The IKEEX…	Started	Automatic	Local System
Interactive Service…	Enables us…	Started	Manual	Local System
Internet Connectio…	Provides n…		Disabled	Local System
IP Helper	Provides tu…	Started	Automatic	Local System

Figure 64. GINA service running as administrator.

As an administrator, the team was able to execute the runcommand.bat file, make a full SSH connection, and update IPTables with threat IPs sent from the RAMPART system. Once the connection was established, the system was setup be continuously updated with threat IP’s. The system performed consistently and robustly as designed, and didn’t experience latency and other issues until the team conducted stress tests.

LIST OF REFERENCES

[1] R. Kelly. (2014). Automated Cyber Threat Analysis and Specified Process Using Vector Relational Data Modeling. Master's Thesis, Naval Postgraduate School.

[2] SANS Institute. Intrusion detection FAQ: What is a honeypot? July 2012. [Online]. Available: http://www.sans.org/security-resources/idfaq/honeypot3.php

[3] U.S. Department of Defense. *Department of Defense Strategy for Operating in Cyberspace*. Washington, D.C.: Dept. of Defense, 2011.

[4] D. Mahon. (2012, Mar.). Testimony before the Subcommittee on Communications and Internet Committee on Energy and Commerce, U.S. House of Representatives. [Online]. Available: http://democrats.energycommerce.house.gov/sites/default/files/documents/Testimony-Mahon-CT-Cybersecurity-Role-of-Communication-Networks-2012-3-7.pdf

[5] J. P. Farwell. (2012). "Industry's vital role in national cyber security." *Strategic Studies Quarterly*. [Online]. *6(4)*, pp. 10–41. Available: http://www.au.af.mil/au/ssq/2012/winter/winter12.pdf

[6] K. Kent and M. Souppaya. (2006, Sep.). NIST SP 800–92, Guide to computer security log management. [Online]. Available: http://csrc.nist.gov/publications/nistpubs/800-92/SP800-92.pdf

[7] S. Gupta. (2012, July). Logging and monitoring to detect network intrusions and compliance violations in the environment. [Online]. Available from SANS Inst. InfoSec Reading Room at http://www.sans.org/reading-room

[8] M. Rouse. (2012, Oct.). Security Information Event Management (SIEM). [Online]. Available: http://searchsecurity.techtarget.com/definition/security-information-and-event-management-SIEM

[9] J. Flynn. (2012). Intrusion along the kill chain. *Black Hat*. [Online]. https://www.blackhat.com/hl/bh-us-12/bh-us-12-archives.hl

[10] Office of the Under Secretary of Defense. *Systems Engineering Guide of Systems*. Washington, D.C.: Dept. of Defense, 2008.

[11] J. Holt, S. Perry, and M. Brownsword, *Model-based Requirements Engineering for systems of systems.* IET Publishing, 2011.

[12] A. Mostafavi, D.M. Abraham, D. DeLaurentis, and J. Sinfield. (2011). "Exploring the dimensions of systems of innovation analysis: A system of systems framework." *IEEE Syst. J.* 5(2), pp. 256–265.

[13] T. Anderson, S. A. McKenzie, C. L. Blais, and D. Brutzman. (2014). "Geospatial mapping of Internet protocol addresses for real-time cyber domain visual analytics and knowledge management using the Global Information Network Architecture." *Nat. Cybersecurity Inst. J. 1(1)*, pp. 33–50.

[14] T. Anderson, D. Dolk, and F. Busalacchi. (2012, Oct.). "Network certification of GINA and development of CONOPs for the GINA tunnel system." *TRADOC Analysis Center Technical Rep.,TRAC-M-TR-13-003*.

[15] D. Dolk, T. Anderson, F. Busalacchi, and D. Tinley. (2012). "GINA: System interoperability for enabling smart mobile system services in network decision support systems." *45th Hawaii Int. Conference on Syst, Sci. (HICSS)*, pp. 1472–1481.

[16] Creek Technologies, LLC. GINA technical white paper – The solution to providing truly integrated solutions to large scale interoperability. [Online]. Available: http://www.creek-technologies.com/sites/default/files/documents/files/GINA%20Technical%20White%20Paper%20-%20Understanding%20and%20Interoperability.pdf

[17] Intelligence Support Organization, *Dragon Pulse Information Management System GINA Manual,* Ft. Leavenworth, KS: Training and Doctrine Command, 2013.

[18] Janalta Interactive. Security – brute force attack. [Online]. Available: http://www.techopedia.com/definition/18091/brute-force-attack

[19] Open Web Application Security Project. Page – brute force attack. [Online]. Available: https://www.owasp.org/index.php/Brute_force_attack

[20] Imperva. Glossary – directory traversal. [Online]. Available: http://www.imperva.com/Resources/Glossary?term=directory_traversal

[21] PuTTY. *Using the command-line connection tool Plink*. [Online]. Available: http://the.earth.li/~sgtatham/putty/0.58/hldoc/Chapter7.hl

[22] Red Hat. *Cygwin user's guide*. [Online]. Available: https://cygwin.com/cygwin-ug-net/cygwin-ug-net.pdf

[23] D. Libes, *Exploring Expect: A Tcl-Based Toolkit for Automating Interactive Programs*. Sebastopol: O'Reilly & Associates, 1995.

[24] Microsoft Corporation. How to change the listening port for remote desktop. [Online]. Available: http://support.microsoft.com/kb/306759

[25] G. Langford. (2014, Feb.). "GINA network-centric assemble-to-description architecture." *Calhoun Inst. Archive of Naval Postgraduate School*. [Online]. Available: http://calhoun.nps.edu/handle/10945/40315

[26] K. A. Stewart. (2013, July). "Innovative network architecture applied to maritime security experimentation." *Naval Postgraduate School*. [Online]. Available: http://www.nps.edu/About/News/Innovative-Network-Architecture-Applied-to-Maritime-Security-Experimentation.hl

[27] T. S. Anderson and F. Busalacchi. (2013, Jan.). "Understanding interoperability and the GINA advantage." [Online]. Available: http://www.acme-tensor.com/uploads/1/6/1/6/16162844/_interoperability_-_gina_advantage.pdf

[28] F. Busalacchi, "Global Information Network Architecture," U.S. Patent 20100070504, Mar 18, 2010.

[29] Mandiant. (2014). *2014 Threat Report: Trends beyond the breach*. [Online]. Available: https://dl.mandiant.com/EE/library/WP_M-Trends2014_140409.pdf

[30] Trustwave Holdings, Inc. (2014). *2014 Trustwave Global Security Report*. [Online]. Available: http://www2.trustwave.com/rs/trustwave/images/2014_Trustwave_Global_Security_Report.pdf

[31] Verizon. (2014). *2014 Data Breach Investigations Report*. [Online]. Available: http://www.verizonenterprise.com/DBIR/2014/

[32] J. Beechey. (2012, Feb.). SIEM based intrusion detection with Q1Labs Qradar. *SANS Inst. InfoSec Reading Room*. [Online]. Available: http://www.sans.org/reading-room/whitepapers/detection/siem-based-intrusion-detection-q1labs-qradar-33278

[33] Verizon. (2013). *Executive Summary 2013 Data Breach Investigations Report.* [Online]. Available: http://www.verizonenterprise.com/resources/insights/detail/133885/2013-Data-Breach-Investigations-Report-Executive-Summary.hl

[34] D. Jewell, R. D. Barros, S. Diederichs, L. M. Duijvestijn, M. Hammersley, A. Hazra, C. Holban, Y. Li, O. Osaigbovo, A. Plach, I. Portilla, M. Saptarshi, H. P. Seera, E. Stahl, and C. Zolotow. (2014, Jan.). Performance and capacity implications for big data. *Int. Bus. Mach.*. [Online]. Available: http://www.redbooks.ibm.com/redpapers/pdfs/redp5070.pdf

[35] I. Linkov and T. P. Seager. *Coupling Multi-Criteria Decision Analysis, Life-Cycle Assessment, and Risk Assessment for Emerging Threats*. American Chemical Society, 2011.

[36] *Multi-criteria Analysis: A Manual*, Dept. for Communities and Local Government Publications, Bressenden Place, UK, 2009.

[37] V. Belton and T. J. Stewart. *Multiple Criteria Decision Analysis: An Integrated Approach*. Boston: Kluwer Academic, 2002.

[38] *Field Manual 6–01.1. Knowledge Management Operations*. Dept. of Army. Washington, D.C., July 16, 2012.

[39] M. Alavi and D.E. Leidner. (2001). "Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues." *MIS Quart. 25(1*), pp. 107–136.

INITIAL DISTRIBUTION LIST

1. Defense Technical Information Center
   Ft. Belvoir, Virginia

2. Dudley Knox Library
   Naval Postgraduate School
   Monterey, California